(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2018년05월31일
(11) 등록번호 10-1863174
(24) 등록일자 2018년05월25일

(51) 국제특허분류(Int. Cl.)
G06F 21/56 (2013.01) *G06F 12/10* (2016.01)
G06F 12/14 (2006.01) *G06F 21/53* (2013.01)
G06F 21/62 (2013.01) *H04L 29/06* (2006.01)
(52) CPC특허분류
G06F 21/566 (2013.01)
G06F 12/109 (2013.01)
(21) 출원번호 10-2015-7022480
(22) 출원일자(국제) 2014년02월04일
심사청구일자 2018년02월05일
(85) 번역문제출일자 2015년08월19일
(65) 공개번호 10-2015-0118957
(43) 공개일자 2015년10월23일
(86) 국제출원번호 PCT/RO2014/000006
(87) 국제공개번호 WO 2014/129918
국제공개일자 2014년08월28일
(30) 우선권주장
13/774,720 2013년02월22일 미국(US)
(56) 선행기술조사문헌
KR101044173 B1
JP2008505400 A
US20070055837 A1
US20090307705 A1

(73) 특허권자
비트데펜더 아이피알 매니지먼트 엘티디
사이프러스 니코시아 1076 12 피시 크레온토스
(72) 발명자
루차스, 안드레이-블라드
루마니아 주데츠 사투 마레, 사투 마레, 불레바르둘 클로슈카 엔알. 111
루칵스, 산도르
루마니아 주데츠 클루지, 사트 플로레슈티, 이티. 3, 불레바르둘 체타테아 페테이 비엘. 비
루차슈, 단-호레아
루마니아 주데츠 클루지, 클루지-나포카, 에이피. 29, 스트라다 호레아 엔알. 89-95
(74) 대리인
권영준

전체 청구항 수 : 총 29 항 심사관 : 윤혜숙

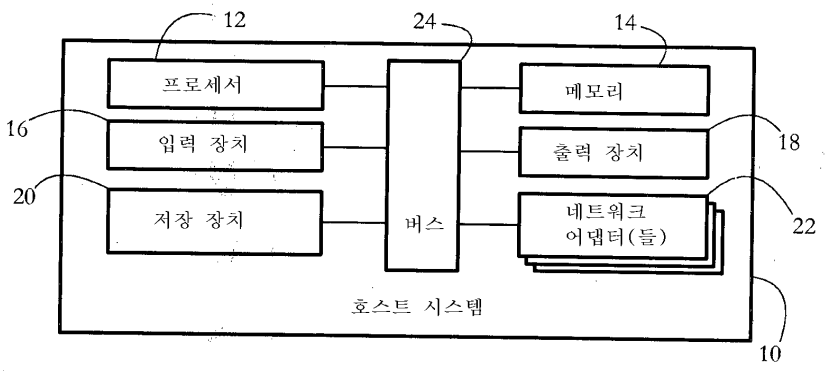
(54) 발명의 명칭 **가상 머신의 일체성 보호를 위한 메모리 인트로스펙션 엔진**

(57) 요 약

본 발명의 시스템 및 방법은 바이러스 및 루트킷과 같은 멀웨어로부터 컴퓨터 시스템을 보호하는 것을 허용한다. 일부 실시예들에서, 하이퍼바이저는 운영 시스템(OS)의 세트를 호스팅하는 하드웨어 가상화 플랫폼을 구성한다. 하이퍼바이저의 프로세서 권한 레벨에서 실행되는 메모리 인트로스펙션 엔진 각각의 OS를 동적으로 식별하고, 메

(뒷면에 계속)

대 표 도 - 도1



모리가 개별 OS에 대해 네이티브한 메모리 할당 함수에 의해 타겟 소프트웨어 오브젝트에 할당되는 방식을 변경하기 위해 보호 프라이밍 모듈을 사용한다. 일부 실시예들에서, 이러한 변경은 멀웨어 보호를 필요로 하는 타겟 오브젝트들에게만 영향을 미치며, 타겟 오브젝트의 데이터를 포함하는 메모리 페이지들이 개별 오브젝트를 위해 독점적으로 예약되게 강제하는 것을 포함한다. 메모리 인트로스펙션 엔진은 개별 메모리 페이지에 대한 쓰기-보호를 제공한다.

(52) CPC특허분류
G06F 12/1466 (2013.01)
G06F 12/1491 (2013.01)
G06F 21/53 (2013.01)
G06F 21/56 (2013.01)
G06F 21/562 (2013.01)
G06F 21/6227 (2013.01)
H04L 63/1441 (2013.01)

명 세 서

청구범위

청구항 1

운영 시스템 및 보호 프라이밍 모듈을 실행하도록 구성된 적어도 하나의 프로세서를 포함하는 호스트 시스템이며,

상기 운영 시스템은 가상 머신의 가상화된 물리적 메모리의 섹션을 가상 머신 내에서 실행되는 타겟 소프트웨어 오브젝트에 할당하도록 구성되고, 가상 머신은 호스트 시스템 상에서 실행되는 하이퍼바이저에 의해 노출되며, 가상화된 물리적 메모리는 페이지들로 구획되고, 하나의 페이지는 호스트 시스템의 물리적 메모리와 가상화된 물리적 메모리 사이에서 개별적으로 맵핑된 메모리의 최소 단위이고,

상기 보호 프라이밍 모듈은 타겟 소프트웨어 오브젝트가 멀웨어 보호를 위한 선택 기준을 만족하는지에 대한 결정에 응답하여, 타겟 소프트웨어 오브젝트가 상기 선택 기준을 만족할 때, 타겟 오브젝트의 메모리 할당을 변경하도록 구성되고, 메모리 할당을 변경하는 것은 타겟 소프트웨어 오브젝트의 적어도 일부를 포함하는 임의의 페이지가 타겟 소프트웨어 오브젝트를 위해 예약되는 것을 보장하는 것을 포함하는, 호스트 시스템.

청구항 2

제1항에 있어서,

보호 프라이밍 모듈에 연결된 메모리 인트로스펙션 엔진을 더 포함하고,

상기 메모리 인트로스펙션 엔진은, 메모리 할당을 변경하는 보호 프라이밍 모듈에 응답하여, 타겟 소프트웨어 오브젝트의 적어도 일부를 포함하는 모든 페이지를 쓰기-방지하도록 구성되는, 호스트 시스템.

청구항 3

제2항에 있어서,

메모리 인트로스펙션 엔진은,

타겟 소프트웨어 오브젝트가 초기화되었는지를 결정하도록, 그리고,

이에 응답하여, 타겟 소프트웨어 오브젝트가 초기화되었을 때, 타겟 소프트웨어 오브젝트의 적어도 일부를 포함하는 모든 페이지를 쓰기-방지하도록 또한 구성되는, 호스트 시스템.

청구항 4

제1항에 있어서, 상기 선택 기준은 운영 시스템의 유형에 따라 타겟 소프트웨어 오브젝트를 선택하는 것을 포함하는, 호스트 시스템.

청구항 5

제4항에 있어서, 타겟 소프트웨어 오브젝트가 멀웨어 보호를 위한 선택 기준을 만족하는지에 대한 결정은 가상 머신의 모델 특이적 레지스터(MSR)의 콘텐츠에 따라 운영 시스템의 유형을 식별하는 것을 포함하는, 호스트 시스템.

청구항 6

제4항에 있어서, 타겟 소프트웨어 오브젝트가 멀웨어 보호를 위한 선택 기준을 만족하는지에 대한 결정은 가상 머신의 모델 특이적 레지스터(MSR)에 의해 포인팅되는 메모리의 콘텐츠에 따라 운영 시스템의 유형을 식별하는 것을 포함하는, 호스트 시스템.

청구항 7

제1항에 있어서, 메모리 할당을 변경하는 것은 운영 시스템의 메모리 할당 함수를 훅킹하는 것을 포함하는, 호

스트 시스템.

청구항 8

제1항에 있어서, 메모리 할당을 변경하는 것은 운영 시스템의 메모리 할당 함수에게 명령하여, 타겟 소프트웨어 오브젝트의 적어도 일부를 포함하는 임의의 페이지를 타겟 소프트웨어 오브젝트에 독점적으로 할당하게 하는, 호스트 시스템.

청구항 9

제8항에 있어서, 메모리 할당 함수에게 명령하는 것은 타겟 소프트웨어 오브젝트의 크기를 페이지 크기의 정수배로 변경하는 것을 포함하는, 호스트 시스템.

청구항 10

제9항에 있어서, 메모리 할당 함수에게 명령하는 것은 상기 섹션을 페이지 바운더리에 정렬시키는 것을 포함하는, 호스트 시스템.

청구항 11

제1항에 있어서, 상기 보호 프라이밍 모듈은 페이지의 예약된 풀을 수립하도록 또한 구성되고, 상기 풀은 멀웨어-보호된 소프트웨어 오브젝트에 대한 할당을 위해 예약되며, 메모리 할당을 변경하는 것은 메모리 페이지의 예약된 풀 내의 섹션을 할당하는 것을 포함하는, 호스트 시스템.

청구항 12

제1항에 있어서,

타겟 소프트웨어 오브젝트가 멀웨어 보호를 위한 선택 기준을 만족하는지에 대한 결정은,

타겟 소프트웨어 오브젝트가 드라이버 오브젝트인지를 결정하는 것과,

이에 응답하여, 타겟 소프트웨어 오브젝트가 드라이버 오브젝트일 때, 타겟 소프트웨어 오브젝트가 멀웨어 보호를 위한 선택 기준을 만족하는지를 결정하는 것을 포함하는, 호스트 시스템.

청구항 13

제1항에 있어서,

상기 보호 프라이밍 모듈은 타겟 오브젝트의 반환을 변경하도록 또한 구성되고, 상기 반환을 변경하는 것은

타겟 소프트웨어 오브젝트의 적어도 일부를 포함하는 페이지가 쓰기-보호되어 있는지를 결정하는 것과,

이에 응답하여, 페이지가 쓰기-보호되어 있을 때, 페이지로부터 쓰기-보호를 제거하는 것을 포함하는, 호스트 시스템.

청구항 14

제12항에 있어서, 반환을 변경하는 것은 운영 시스템의 메모리 반환 함수를 훅킹하는 것을 더 포함하는, 호스트 시스템.

청구항 15

운영 시스템을 형성하기 위해 호스트 시스템의 적어도 하나의 프로세서를 사용하는 단계로서, 상기 운영 시스템은 가상 머신의 가상화된 물리적 메모리의 섹션을 가상 머신 내에서 실행되는 타겟 소프트웨어 오브젝트에 할당하도록 구성되고, 가상 머신은 호스트 시스템 상에서 실행되는 하이퍼바이저에 의해 노출되며, 가상화된 물리적 메모리는 페이지들로 구획되고, 하나의 페이지는 호스트 시스템의 물리적 메모리와 가상화된 물리적 메모리 사이에서 개별적으로 맵핑된 메모리의 최소 단위인, 운영 시스템을 형성하기 위해 호스트 시스템의 적어도 하나의 프로세서를 사용하는 단계와,

타겟 소프트웨어 오브젝트가 멀웨어 보호를 위한 선택 기준을 만족하는지에 대한 결정에 응답하여, 타겟 소프트웨어 오브젝트가 상기 선택 기준을 만족할 때, 타겟 소프트웨어 오브젝트의 메모리 할당을 변경하도록 상기 적

어도 하나의 프로세서를 사용하는 단계를 포함하며,

메모리 할당을 변경하는 단계는 타겟 소프트웨어 오브젝트의 적어도 일부를 포함하는 임의의 페이지가 타겟 소프트웨어 오브젝트를 위해 예약되는 것을 보장하는 단계를 포함하는, 방법.

청구항 16

제15항에 있어서, 메모리 할당을 변경하는 단계에 응답하여, 타겟 소프트웨어 오브젝트의 적어도 일부를 포함하는 모든 페이지를 쓰기-방지하는 단계를 더 포함하는, 방법.

청구항 17

제16항에 있어서,

타겟 소프트웨어 오브젝트가 초기화되었는지를 결정하는 단계와,

이에 응답하여, 타겟 소프트웨어 오브젝트가 초기화되었을 때, 타겟 소프트웨어 오브젝트의 적어도 일부를 포함하는 모든 페이지를 쓰기-방지하는 단계를 더 포함하는, 방법.

청구항 18

제15항에 있어서, 상기 선택 기준은 운영 시스템의 유형에 따라 타겟 소프트웨어 오브젝트를 선택하는 단계를 포함하는, 방법.

청구항 19

제18항에 있어서, 타겟 소프트웨어 오브젝트가 멀웨어 보호를 위한 선택 기준을 만족하는지에 대한 결정은 가상 머신의 모델 특이적 레지스터(MSR)의 콘텐츠에 따라 운영 시스템의 유형을 식별하는 것을 포함하는, 방법.

청구항 20

제18항에 있어서, 타겟 소프트웨어 오브젝트가 멀웨어 보호를 위한 선택 기준을 만족하는지에 대한 결정은 가상 머신의 모델 특이적 레지스터(MSR)에 의해 포인팅되는 메모리의 콘텐츠에 따라 운영 시스템의 유형을 식별하는 것을 포함하는, 방법.

청구항 21

제15항에 있어서, 메모리 할당을 변경하는 단계는 운영 시스템의 메모리 할당 함수를 훅킹하는 단계를 포함하는, 방법.

청구항 22

제21항에 있어서, 메모리 할당을 변경하는 단계는 타겟 소프트웨어 오브젝트의 적어도 일부를 포함하는 모든 페이지를 타겟 소프트웨어 오브젝트에 독점적으로 할당하도록, 운영 시스템의 메모리 할당 함수에게 명령하는 단계를 포함하는, 방법.

청구항 23

제22항에 있어서, 메모리 할당 함수에게 명령하는 단계는 타겟 소프트웨어 오브젝트의 크기를 페이지 크기의 정수배로 변경하는 단계를 포함하는, 방법.

청구항 24

제23항에 있어서, 메모리 할당 함수에게 명령하는 단계는 상기 섹션을 페이지 바운더리에 정렬시키는 단계를 포함하는, 방법.

청구항 25

제15항에 있어서, 메모리 할당을 변경하는 단계는

페이지의 예약된 풀을 수립하는 단계로서, 상기 풀은 멀웨어-보호된 소프트웨어 오브젝트에 대한 할당을 위해 예약되는, 페이지의 예약된 풀을 수립하는 단계와,

메모리 페이지의 예약된 풀 내의 섹션을 할당하는 단계를 포함하는, 방법.

청구항 26

제15항에 있어서, 타겟 소프트웨어 오브젝트가 멀웨어 보호를 위한 선택 기준을 만족하는지에 대한 결정은,

타겟 소프트웨어 오브젝트가 드라이버 오브젝트인지를 결정하는 단계와,

이에 응답하여, 타겟 소프트웨어 오브젝트가 드라이버 오브젝트일 때, 타겟 소프트웨어 오브젝트가 멀웨어 보호를 위한 선택 기준을 만족하는지를 결정하는 단계를 포함하는, 방법.

청구항 27

제15항에 있어서, 타겟 소프트웨어 오브젝트의 반환을 변경하기 위해 적어도 하나의 프로세서를 사용하는 단계를 더 포함하고, 반환을 변경하는 단계는

소프트웨어 오브젝트의 적어도 일부를 포함하는 페이지가 쓰기-보호되어 있는지를 결정하는 단계와,

이에 응답하여, 페이지가 쓰기-보호되어 있을 때, 페이지로부터 쓰기-보호를 제거하는 단계를 포함하는, 방법.

청구항 28

제27항에 있어서, 반환을 변경하는 단계는 운영 시스템의 메모리 반환 함수를 훅킹하는 단계를 더 포함하는, 방법.

청구항 29

호스트 시스템의 적어도 하나의 프로세서에 의해 실행될 때, 상기 적어도 하나의 프로세서가,

가상 머신의 가상화된 물리적 메모리의 섹션을 가상 머신 내에서 실행되는 타겟 소프트웨어 오브젝트에 할당하게, 그리고

타겟 소프트웨어 오브젝트가 멀웨어 보호를 위한 선택 기준을 만족하는지에 대한 결정에 응답하여, 타겟 소프트웨어 오브젝트가 상기 선택 기준을 만족할 때, 타겟 소프트웨어 오브젝트의 메모리 할당을 변경하게, 하는 명령을 인코딩하는 비-일시적 컴퓨터 판독가능 매체이며,

가상 머신은 호스트 시스템 상에서 실행되는 하이퍼바이저에 의해 노출되며, 가상화된 물리적 메모리는 페이지들로 구획되고, 하나의 페이지는 호스트 시스템의 물리적 메모리와 가상화된 물리적 메모리 사이에서 개별적으로 맵핑된 메모리의 최소 단위이고,

메모리 할당을 변경하는 것은 타겟 소프트웨어 오브젝트의 적어도 일부를 포함하는 임의의 페이지가 타겟 소프트웨어 오브젝트를 위해 예약되는 것을 보장하는 것을 포함하는, 비-일시적 컴퓨터 판독가능 매체.

발명의 설명

기 술 분 야

[0001] 본 발명은 멀웨어(malware)로부터 컴퓨터 시스템을 보호하기 위한 시스템 및 방법, 특히 하드웨어 가상화 기술을 사용하는 안티-멀웨어 시스템에 관한 것이다.

배 경 기 술

[0002] 멀웨어로도 알려진 악성 소프트웨어는 세계적으로 상당수의 컴퓨터 시스템에 영향을 주고 있다. 멀웨어는 컴퓨터 바이러스, 웜, 및 루트킷(rootkit)과 같은 많은 형태로, 수백만의 컴퓨터 사용자에게 심각한 위협이 되고 있으며, 무엇보다도 데이터 및 민감한 정보의 손실, 신원 도용, 및 생산성 손실에 있어 이들을 취약하게 하고 있다.

[0003] 하드웨어 가상화 기술은, 많은 방면에서 물리적 컴퓨터 시스템으로 거동하는 통상적으로 가상 머신(virtual machine)으로 알려진 시뮬레이티드 컴퓨터 환경의 형성을 허용한다. 서버 통합 및 서비스로서의 인프라스트럭쳐(IAAS)와 같은 통상적인 어플리케이션에서, 몇몇 가상 머신은 동일한 물리적 머신상에서 동시에 실행될 수 있어서, 하드웨어 자원을 그들 사이에서 공유함으로써, 투자 및 운영 비용을 줄일 수 있다. 각각의 가상 머신은

다른 가상 머신들과 별개로 고유의 운영 시스템 및/또는 소프트웨어 어플리케이션을 구동할 수 있다. 멀웨어의 지속적인 확산으로 인해, 이러한 환경에서 작동하는 각각의 가상 머신은 잠재적으로 멀웨어 보호를 필요로 한다.

[0004] 본 기술분야에서 통상적으로 사용되는 가상화 솔루션은 가상 머신 모니터로도 알려진 하이퍼바이저(hypervisor)를 포함하며, 이는 가상 머신의 운영 시스템(OS)과 컴퓨팅 하드웨어 사이에서 작동하는 소프트웨어의 레이어로 구성되고, 개별 OS보다 더 많은 프로세서 권한(processor privilege)을 갖는다. 안티-멀웨어 작동은 하이퍼바이저의 권한 레벨에서 수행될 수 있다. 이러한 구성이 멀웨어 검출 및 방지를 촉진할 수는 있겠지만, 이들은 복잡한 엑스트라 레이어를 도입하며, 상당한 연산 비용을 수반할 수 있다.

[0005] 최소한의 연산 고정비(overhead)로, 하드웨어 가상화 플랫폼을 위한, 로버스트(robust)하고 스케일가능한 안티-멀웨어 솔루션을 개발하고자 하는 상당한 관심이 존재한다.

발명의 내용

해결하려는 과제

과제의 해결 수단

[0006] 일 양태에 따르면, 호스트 시스템은 운영 시스템 및 보호 프라이밍 모듈을 실행하도록 구성된 적어도 하나의 프로세서를 포함한다. 상기 운영 시스템은 가상 머신의 가상화된 물리적 메모리의 섹션을 가상 머신 내에서 실행되는 타겟 소프트웨어 오브젝트에 할당하도록 구성되고, 가상 머신은 호스트 시스템 상에서 실행되는 하이퍼바이저에 의해 노출된다. 가상화된 물리적 메모리는 페이지들로 구획되고, 하나의 페이지는 호스트 시스템의 물리적 메모리와 가상화된 물리적 메모리 사이에서 개별적으로 맵핑된 메모리의 최소 단위이다. 상기 보호 프라이밍 모듈은 타겟 소프트웨어 오브젝트가 멀웨어 보호를 위한 선택 기준을 만족하는지에 대한 결정에 응답하여, 타겟 소프트웨어 오브젝트가 상기 선택 기준을 만족할 때, 타겟 오브젝트의 메모리 할당을 변경하도록 구성되고, 메모리 할당을 변경하는 것은 타겟 소프트웨어 오브젝트의 적어도 일부를 포함하는 임의의 페이지가 타겟 소프트웨어 오브젝트를 위해 예약되는 것을 보장하는 것을 포함한다.

[0007] 다른 양태에 따르면, 방법은 운영 시스템을 형성하기 위해 호스트 시스템의 적어도 하나의 프로세서를 사용하는 단계로서, 상기 운영 시스템은 가상 머신의 가상화된 물리적 메모리의 섹션을 가상 머신 내에서 실행되는 타겟 소프트웨어 오브젝트에 할당하도록 구성되고, 가상 머신은 호스트 시스템 상에서 실행되는 하이퍼바이저에 의해 노출되며, 가상화된 물리적 메모리는 페이지들로 구획되고, 하나의 페이지는 호스트 시스템의 물리적 메모리와 가상화된 물리적 메모리 사이에서 개별적으로 맵핑된 메모리의 최소 단위인, 운영 시스템을 형성하기 위해 호스트 시스템의 적어도 하나의 프로세서를 사용하는 단계를 포함한다. 방법은 타겟 소프트웨어 오브젝트가 멀웨어 보호를 위한 선택 기준을 만족하는지에 대한 결정에 응답하여, 타겟 소프트웨어 오브젝트가 상기 선택 기준을 만족할 때, 타겟 소프트웨어 오브젝트의 메모리 할당을 변경하도록 상기 적어도 하나의 프로세서를 사용하는 단계를 더 포함하고, 메모리 할당을 변경하는 단계는 타겟 소프트웨어 오브젝트의 적어도 일부를 포함하는 임의의 페이지가 타겟 소프트웨어 오브젝트를 위해 예약되는 것을 보장하는 단계를 포함한다.

[0008] 또 다른 양태에 따르면, 비-일시적 컴퓨터 판독가능 매체는 호스트 시스템의 적어도 하나의 프로세서에 의해 실행될 때, 상기 적어도 하나의 프로세서가, 가상 머신의 가상화된 물리적 메모리의 섹션을 가상 머신 내에서 실행되는 타겟 소프트웨어 오브젝트에 할당하게 하는 명령을 인코딩하고, 가상 머신은 호스트 시스템 상에서 실행되는 하이퍼바이저에 의해 노출된다. 가상화된 물리적 메모리는 페이지들로 구획되고, 하나의 페이지는 호스트 시스템의 물리적 메모리와 가상화된 물리적 메모리 사이에서 개별적으로 맵핑된 메모리의 최소 단위이다. 상기 명령은 또한, 상기 적어도 하나의 프로세서가, 타겟 소프트웨어 오브젝트가 멀웨어 보호를 위한 선택 기준을 만족하는지에 대한 결정에 응답하여, 타겟 소프트웨어 오브젝트의 메모리 할당을 변경하게 한다. 메모리 할당의 변경은 타겟 소프트웨어 오브젝트가 상기 선택 기준을 만족할 때 실행된다. 메모리 할당을 변경하는 것은 타겟 소프트웨어 오브젝트의 적어도 일부를 포함하는 임의의 페이지가 타겟 소프트웨어 오브젝트를 위해 예약되는 것을 보장하는 것을 포함한다.

도면의 간단한 설명

[0009] 본 발명의 전술한 양태들 및 장 점은 후술하는 상세한 설명 및 도면을 참조로 읽을 때 더욱 잘 이해될

것이다.

도 1은 본 발명의 일부 실시예들에 따른, 멀웨어로부터 보호되는 호스트 컴퓨터 시스템의 예시적인 하드웨어 구성을 도시한다.

도 2는 본 발명의 일부 실시예들에 따른, 도 1의 호스트 시스템상에서 실행되는 하이퍼바이저에 의해 노출된 가상 머신의 예시적인 세트를 도시한다.

도 3은 본 발명의 일부 실시예들에 따른, 멀웨어로부터 보호되는 오브젝트의 세트를 포함하는, 다양한 프로세서 권한 레벨로 호스트 시스템상에서 실행되는 소프트웨어 오브젝트의 예시적인 계층(hierarchy)을 도시한다.

도 4는 본 발명의 일부 실시예들에 따른, 도 2의 시스템 구성 내의 메모리 어드레스들의 예시적인 맵핑을 도시한다.

도 5는 예시적인 메모리 할당을 도시하며, 가상화된 물리적 메모리 공간은 페이지들로 분할되며, 메모리의 구분된 섹션은 복수의 소프트웨어 오브젝트의 각각에 할당된다.

도 6은 예시적인 메모리 할당을 도시하며, 한 페이지가 본 발명의 일부 실시예들에 따라 보호되는 소프트웨어 오브젝트에 독점적으로 할당된다.

도 7은 본 발명의 일부 실시예들에 따른, 가상 머신을 보호하기 위해 메모리 인트로스펙션 엔진(memory introspection engine)에 의해 실행되는 단계들의 예시적인 시퀀스를 도시한다.

도 8은 타겟 소프트웨어 오브젝트에 대한 메모리 할당을 수정하기 위해 보호 프라이밍 모듈(protection priming module)의 실시예에 의해 실행되는 단계들의 예시적인 시퀀스를 도시한다.

도 9는 타겟 소프트웨어 오브젝트에 대한 메모리 할당을 수정하기 위해 보호 프라이밍 모듈의 다른 실시예에 의해 실행되는 단계들의 예시적인 시퀀스를 도시한다.

도 10은 본 발명의 일부 실시예들에 따른, 타겟 소프트웨어 오브젝트에 대한 메모리 반환(de-allocation)을 수정하기 위해 보호 프라이밍 모듈에 의해 실행되는 단계들의 예시적인 시퀀스를 도시한다.

발명을 실시하기 위한 구체적인 내용

[0010] 이하의 설명에서, 구조들 사이에서 언급된 모든 연결들은 직접적인 동작 연결들 또는 매개 구조들을 통한 간접적인 동작 연결들일 수 있는 것으로 이해된다. 구성 요소들의 세트는 하나 이상의 구성 요소를 포함한다. 구성 요소의 임의의 열거는 적어도 하나의 구성 요소를 언급하는 것으로 이해된다. 복수의 구성 요소는 적어도 2개의 구성 요소를 포함한다. 달리 요구되지 않는다면, 기술된 어떠한 방법 단계들도 설명된 특정 순서로 반드시 실행될 필요는 없다. 제2 구성 요소로부터 유도되는 제1 구성 요소(예컨대, 데이터)는 제2 구성 요소와 동일한 제1 구성 요소는 물론, 제2 구성 요소 그리고 선택적으로는 다른 데이터를 처리하는 것에 의해 생성된 제1 구성 요소를 포함한다. 파라미터에 따라 결정 또는 판정하는 것은 파라미터에 따라 그리고 선택적으로는 다른 데이터에 따라 결정 또는 판정하는 것을 포함한다. 달리 구체화되지 않는다면, 일부 수량/데이터의 인디케이터는 수량/데이터 그 자체, 또는 수량/데이터 그 자체와 상이한 인디케이터일 수 있다. 달리 구체화되지 않는다면, 페이지는 호스트 시스템의 가상 메모리로 개별적으로 맵핑된 가상화된 물리적 메모리의 최소 단위를 나타낸다. 달리 구체화되지 않는다면, 가상화된 물리적 메모리의 섹션은, 페이지 세트의 각각의 페이지가 개별 섹션의 일부를 포함할 때, 스팬(span)한다고 말한다. 타겟 오브젝트를 위해 페이지를 예약한다는 것은 전체 페이지를 타겟 오브젝트에 할당하는 것, 그렇지 않으면, 개별 페이지가 타겟 오브젝트와 구분되는 오브젝트의 부분들을 호스팅하지 않는 것을 보장하는 것을 포함한다. 컴퓨터 판독 가능 매체는 자성, 광, 및 반도체 저장 매체(예컨대, 하드 드라이브, 광 디스크, 플래시 메모리, DRAM)와 같은 비-일시적 매체(non-transitory medium)는 물론, 전도성 케이블 및 파이버 옵틱 링크와 같은 통신 링크들을 포함한다. 일부 실시예들에 따르면, 본 발명은, 그 중에서도, 본원에 설명된 방법들을 수행하기 위해 프로그래밍된 하드웨어(예컨대, 하나 이상의 프로세서)는 물론, 본원에서 설명된 방법들을 수행하기 위한 명령들을 인코딩하는 컴퓨터-판독 가능 매체를 포함하는 컴퓨터 시스템을 제공한다.

[0011] 후술하는 설명은 본 발명의 실시예들을 예시적으로 설명하는 것이며, 반드시 제한적인 것은 아니다.

[0012] 도 1은 본 발명의 일부 실시예들에 따른 호스트 시스템(10)의 예시적인 하드웨어 구성을 도시한다. 호스트 시스템(10)은 특히 엔터프라이즈 서버와 같은 기업용 컴퓨팅 장치, 또는 개인용 컴퓨터나 스마트폰과 같은 엔드-유저 장치를 나타낼 수 있다. 다른 호스트 시스템들은 TV 및 게임 콘솔과 같은 엔터테인먼트 장치, 또는 메모

리와 프로세서를 구비하고, 가상화를 지원하며, 멀웨어 보호를 필요로 하는 임의의 다른 장치를 포함한다. 도 1은 설명을 위한 컴퓨터 시스템을 도시하며, 휴대 전화 또는 태블릿과 같은 다른 클라이언트 장치들은 상이한 구성을 가질 수 있다. 일부 실시예들에서, 시스템(10)은 버스(24)의 세트에 의해 모두 연결되어 있는 프로세서(12), 메모리 유닛(14), 입력 장치(16) 세트, 출력 장치(18) 세트, 저장 장치(20) 세트, 및 네트워크 어댑터(22) 세트를 포함하는 물리적 장치들의 세트를 포함한다.

[0013] 일부 실시예들에서, 프로세서(12)는 신호 및/또는 데이터의 세트로 산술 및/또는 논리 연산을 실행하도록 구성된 물리적 장치(예컨대, 멀티-코어 집적 회로)를 포함한다. 일부 실시예들에서, 이러한 논리 연산들은 프로세서 명령(예를 들어, 기계 코드 또는 다른 유형의 소프트웨어)의 시퀀스 형태로 프로세서(12)에 전달된다. 메모리 유닛(14)은 명령들을 수행하는 도중에 프로세서(12)에 의해 액세스되거나 생성되는 데이터/신호들을 저장하는 휘발성 컴퓨터-판독 가능 매체(예컨대, RAM)를 포함할 수 있다. 입력 장치(16)는 사용자가 시스템(10)으로 데이터 및/또는 명령들을 도입할 수 있게 하는 개별 하드웨어 인터페이스 및/또는 어댑터를 포함하는, 특히 컴퓨터 키보드, 마우스, 및 마이크를 포함할 수 있다. 출력 장치(18)는 특히 모니터와 같은 디스플레이 장치 및 스피커는 물론, 시스템(10)이 사용자에게 데이터를 통신하게 할 수 있는 그래픽 카드와 같은 하드웨어 인터페이스/어댑터를 포함할 수 있다. 일부 실시예들에서, 입력 장치(16)와 출력 장치(18)는 터치-스크린 장치들의 경우와 같이, 하드웨어의 공통적인 부품을 공유할 수 있다. 저장 장치(20)는 소프트웨어 명령들 및/또는 데이터의 비휘발성 저장, 판독, 및 기록을 가능하게 하는 컴퓨터-판독 가능 매체를 포함한다. 예시적인 저장 장치(20)는 자성 및 광 디스크들 및 플래시 메모리 장치들은 물론, CD 및/또는 DVD 디스크들 및 드라이브들과 같은 소거 가능 매체를 포함한다. 네트워크 어댑터(22) 세트는 시스템(10)이 컴퓨터 네트워크 및/또는 다른 장치들/컴퓨터 시스템들에 연결될 수 있게 한다. 버스(24)는 호스트 시스템(10)의 장치들(12-22)의 상호-통신을 가능하게 하는 복수의 시스템, 주변 장치, 및 칩셋 버스들, 및/또는 다른 모든 회로망을 집합적으로 나타낸다. 예를 들어, 버스(24)는 특히 프로세서(12)를 메모리(14)에 연결시키는 노스브리지, 및/또는 프로세서(12)를 장치들(16-22)에 연결시키는 사우스브리지를 포함할 수 있다.

[0014] 도 2는 본 발명의 일부 실시예들에 따른 하이퍼바이저(30)에 의해 노출되고 호스트 시스템(10) 상에서 실행되는 게스트 가상 머신(32a-b)의 예시적인 세트를 도시한다. 가상 머신(VM)은 본 기술분야에서 통상적으로 고유의 운영 시스템 및 다른 VM과 무관한 소프트웨어를 각각 구동할 수 있는 실제 물리적 머신/컴퓨터 시스템의 소프트웨어 에뮬레이션으로 알려져 있다. 하이퍼바이저(30)는 프로세서 동작, 메모리, 저장소, 입력/출력, 및 네트워킹 장치들과 같은 호스트 시스템(10)의 하드웨어 자원의 다수의 가상 머신에 의해 멀티플렉싱(공유)을 허용하는 소프트웨어를 포함한다. 일부 실시예들에서, 하이퍼바이저(30)는 다수의 가상 머신들 및/또는 운영 시스템(OS)들이 다양한 고립도(degree of isolation)로, 호스트 시스템(10) 상에서 동시에 실행되게 한다. 이러한 구성을 가능하게 하기 위해, 하이퍼바이저(30)의 일부를 구성하는 소프트웨어는 복수의 가상화, 즉 소프트웨어-에뮬레이티드 장치들을 생성할 수 있으며, 각각의 가상화된 장치는 특히 프로세서(12) 및 메모리(14)와 같은 시스템(10)의 물리적 하드웨어 장치를 에뮬레이팅한다. 하이퍼바이저(30)는 호스트 시스템(10) 상에서 작동하는 각각의 VM에 대해 가상 장치들의 세트를 또한 할당할 수 있다. 따라서, 각각의 VM(32a-b)은 고유의 물리적 장치 세트를 구비하는 것처럼, 즉 거의 완벽한 컴퓨터 시스템인 것처럼 작동한다. 유명한 하이퍼바이저의 예로는, 특히 VMware Inc.의 VMware vSphere™ 및 오픈 소스 Xen 하이퍼바이저가 있다.

[0015] 일부 실시예들에서, 하이퍼바이저(30)는 이하에서 더욱 설명되는 바와 같은 안티-멀웨어 작동을 수행하도록 구성된 메모리 인트로스펙션 엔진(40), 및 메모리 인트로스펙션 엔진(40)에 연결된 보호 프라이밍 모듈(46)을 포함한다. 엔진(40) 및 모듈(46)은 하이퍼바이저(30)에 통합되거나, 되지만 하이퍼바이저(30)와는 구별되고 독립적인 소프트웨어 구성요소로서 전달될 수 있지만, 하이퍼바이저(30)와 실질적으로 동일한 프로세서 권한 레벨로 실행된다. 단일 엔진(40)은 호스트 시스템(10) 상에서 실행되는 다수의 멀웨어 보호 VM으로 구성될 수 있다.

[0016] 도 2가 단순화를 위해 단지 2개의 VM(32a-b)을 도시하고 있더라도, 호스트 시스템(10)은 많은 개수의, 예컨대 수백 개의 VM을 작동시킬 수 있으며, 이러한 VM의 개수는 호스트 시스템(10)이 작동하는 동안 변경될 수 있다. 일부 실시예들에서, 각각의 VM(32a-b)은 각각 호스트 시스템(10) 상에서 다른 VM과 독립적으로 그리고 동시에 실행되는 게스트 운영 시스템(34a-b) 및/또는 소프트웨어 어플리케이션(42a-b 및 42c-d) 세트를 실행한다. 각각의 OS(34a-b)는 개별 VM(32a-b)의 가상화된 하드웨어에 대한 인터페이스를 제공하고, 개별 OS 상에서 각각 실행되는 어플리케이션(42a-b 및 42c-d)을 연산하기 위한 호스트로서의 역할을 하는 소프트웨어를 포함한다. 운영 시스템(34a-b)은 특히 Windows®, MacOS®, Linux®, iOS®, 또는 Android®와 같은 널리 입수가능한 운영 시스템을 포함할 수 있다. 어플리케이션(42a-d)은 특히 워드 프로세싱, 화상 처리, 데이터베이스, 브라우저, 전자 통신 어플리케이션, 및 안티-멀웨어 어플리케이션을 포함할 수 있다.

[0017] 도 3은 본 발명의 일부 실시예들에 따른 호스트 시스템(10) 상에서 실행되는 소프트웨어 오브젝트의 계층을 도시한다. 도 3은 본 기술분야에서 레이어 또는 프로텍션 링으로도 알려진, 프로세서 권한 레벨의 관점에서 표현되었다. 일부 실시예들에서, 하이퍼바이저(30)는 가장 권한있는 레벨(통상적으로는 루트 모드, 또는 Intel 플랫폼에서의 VMXroot로 알려짐)에서 프로세서(12)의 제어권을 가짐으로써, 호스트 시스템(10) 상에서 실행되는 다른 소프트웨어에 대해 가상 머신(32)으로써 제공되는 하드웨어 가상화 플랫폼을 생성한다. 도 2의 OS(34a-b)와 같은 운영 시스템(34)은 VM(32)의 가상 환경 내에서 실행되며, OS(34)는 하이퍼바이저(30)보다 낮은 프로세서 권한[예컨대, 커널 모드 또는 Intel 플랫폼에서의 링 0(ring 0)]을 갖는다. OS(34)는 OS 권한 모드에서 실행되는 보호된 OS 오브젝트(36a)를 더 포함할 수 있다. 일부 실시예들에서, 보호된 OS 오브젝트(36a)는, 이하에서 추가로 나타나는 바와 같이, 드라이버 오브젝트와 같은 멀웨어 보호를 위해 선택된 데이터 구조를 포함한다. 도 2의 어플리케이션(42a-d)들과 같은 어플리케이션(42e-f) 세트는 OS(34)의 권한 레벨보다 낮은 프로세서 권한 레벨(예컨대, Intel 플랫폼에서의 사용자 모드)에서 실행된다.

[0018] 일부 실시예들에서, 안티-멀웨어 어플리케이션(44)은, 통상적으로 어플리케이션(42e-f)과 동일한 프로세서 권한 레벨로 OS(34) 상에서 실행된다. 예시적인 안티-멀웨어 어플리케이션(44)은 안티-바이러스 소프트웨어 또는 안티-바이러스를 포함하는 보다 큰 소프트웨어 스윗(software suite), 안티-스파이웨어 및 다른 컴퓨터 보안 어플리케이션을 포함한다. 일부 실시예들에서, 안티-멀웨어 드라이버(36b)는 OS(34)의 프로세서 레벨과 유사한 프로세서 레벨로 실행된다. 드라이버(36b)는 안티-멀웨어 어플리케이션(44)에 대해, 예컨대 OS(34) 상에서 실행되는 소프트웨어 오브젝트의 멀웨어-지시 거동의 검출 및/또는 멀웨어 서명에 대한 메모리 스캔을 위한 기능을 제공한다.

[0019] 일부 실시예들에서, 인트로스펙션 엔진(40)은 하이퍼바이저(30)와 실질적으로 동일한 레벨로 실행되고, 가상 머신(32)과 같은 가상 머신의 인트로스펙션을 실행하도록 구성된다. VM 또는 개별 VM 상에서 실행되는 소프트웨어 오브젝트의 인트로스펙션은, 특히 소프트웨어 오브젝트의 거동을 분석하는 것, 그러한 소프트웨어 오브젝트의 메모리 어드레스들을 결정 및/또는 액세스하는 것, 그러한 어드레스들에 위치한 메모리의 콘텐츠에 대한 특정 프로세스의 액세스를 제한하는 것, 및 그러한 콘텐츠를 분석하는 것을 포함할 수 있다.

[0020] 예시적인 작동에서, 메모리 인트로스펙션 엔진(40)은 멀웨어로부터 보호되는 소프트웨어 오브젝트들로 이루어진 보호 영역(38)을 설정할 수 있다. 이러한 오브젝트들을 보호하는 것에는 개별 오브젝트들에 속하는 데이터를 저장하는 메모리 구역에 대한 액세스를 제한하는 것이 포함될 될 수 있다. 하이퍼바이저(30)는 VM(32)의 메모리 공간에 걸쳐 제어권을 갖기 때문에, OS(34)에 의해 사용된 메모리의 특정 구역을 보호하는 것은, 예컨대 개별 메모리 구역들에 대한 적절한 액세스 권한을 설정하는 하이퍼바이저(30)에 의해 달성될 수 있다. 일부 실시예들에서, 보호 영역(38)은 보호된 OS 오브젝트(36a) 및 안티-멀웨어 드라이버(36b)를 포함한다. OS(34)가 Linux 운영 시스템인 경우, 예시적인 보호된 OS 오브젝트(36a)는, 특히 커널(리드-온리 코드 및/또는 sys_call_table과 같은 데이터), sysenter/syscall 제어 레지스터, 및 어드레스 int 0x80 (syscall) 및/또는 int 0x01을 포함한다. Windowos OS의 예시적인 보호된 오브젝트는 커널(리드-온리 코드 및/또는 System Service Dispatch Table을 포함하는 데이터), 다양한 디스크립터 테이블(예컨대, 인터럽트, 제네럴, 및/또는 로컬), sysenter/syscall 제어 레지스터 및/또는 인터럽트 디스크립터 테이블 레지스터(IDTR)와 같은 다른 레지스터, 글로벌 디스크립터 테이블 레지스터(GDTR), 및 로컬 디스크립터 테이블 레지스터(LDTR)를 포함한다. 일부 실시예들에서, 보호된 OS 오브젝트(36a)는, 특히 특이적 드라이버 오브젝트(specific driver object) 및 패스트 I/O 디스패치 테이블(예컨대, disk, atapi, clfs, fltmgr, ntfs, fastfat, iastor, iastorv)을 또한 포함할 수 있다. 다른 보호된 오브젝트(36a)는 ia32_systenter_eip, ia32_sysenter_esp, ia32_efer, ia32_star, ia32_lstar, 및 ia32_gs_base와 같은 특정 모델 특이적 레지스터(MSR)를 또한 포함할 수 있다. 일부 실시예들에서, 인트로스펙션 엔진(40)은 악의적 코드를 수용하는 어드레스들로의 미인가 리라우팅(rerouting)을 방지하기 위해 페이지 테이블을 또한 보호한다.

[0021] 가상 머신들은 가상화된 물리적 메모리로, 즉 호스트 시스템(10)의 실제 물리적 메모리(14)의 가상 리프레젠테이션(virtual representation)으로 작동한다. 가상화된 물리적 메모리는 각각의 게스트 VM(32a-b)에 대해 특이적인, 가상화된 어드레스들의 인접 공간을 포함하며, 상기 공간의 부분들은 물리적 메모리(14) 및/또는 물리적 저장 장치(20) 내의 어드레스로 맵핑된다(mapped). 가상화를 지원하도록 구성된 시스템에서, 이러한 맵핑은 통상적으로 익스텐디드 페이지 테이블(EPT) 또는 네스티드 페이지 테이블(NPT)과 같은 프로세서(12)에 의해 제어되는 전용(dedicated) 데이터 구조에 의해 달성된다. 이러한 시스템에서, 가상화된 물리적 메모리는 본 기술분야에서 페이지로 알려진 유닛들로 구획될 수 있으며, 하나의 페이지는 EPT 및/또는 NPT와 같은 메커니즘을 통해 물리적 메모리로 개별적으로 맵핑된 가상화된 물리적 메모리의 최소 단위를 나타내는데, 이는 즉 물리적 메모리

와 가상화된 물리적 메모리 사이의 맵핑이 페이지 그래뉼러리티(page granularity)로 수행된다는 것이다. 모든 페이지들은 통상적으로 미리 결정된 크기, 예컨대 4 킬로바이트, 2 메가바이트 등의 크기를 갖는다. 가상화된 물리적 메모리의 구획화는 통상적으로 하이퍼바이저(30)에 의해 구성된다. 일부 실시예들에서, 하이퍼바이저(30)는 EPT/NPT 및 그로 인한 물리적 메모리와 가상화된 물리적 메모리 사이의 맵핑을 또한 구성한다. 일부 하드웨어 구성은, 예컨대 개별 페이지에 대한 읽기 및 쓰기 액세스 권한을 설정함으로써, 하이퍼바이저(30)가 각 페이지 내에 저장된 데이터에 대한 액세스를 선택적으로 제어하게 한다. 이러한 권한은 예컨대, EPT 또는 NPT 내의 개별 페이지의 엔트리를 수정함으로써 설정될 수 있다. 이로 인해, 하이퍼바이저(30)는 어느 소프트웨어 오브젝트가 각 페이지 내의 어드레스들에 저장된 데이터를 액세스할지를 선택하고, 개별 데이터로 어떤 작업, 예컨대 읽기, 쓰기 등이 허용되는지를 지시할 수 있다. 오브젝트가 개별 권한을 갖지 않는 페이지로부터 데이터를 읽거나 그 페이지에 데이터를 쓰는 것과 같은 작업을 실행하기 위한 소프트웨어 오브젝트의 시도는 가상 머신 엑시트 이벤트(예컨대, Intel 플랫폼에서의 VMExit 이벤트)를 촉발할 수 있다. 일부 실시예들에서, 가상 머신 엑시트 이벤트는 개별 소프트웨어 오브젝트를 실행하는 VM으로부터 하이퍼바이저(30)로 제어권을 넘김으로써, 미인가 읽기/쓰기 시도를 방지한다.

[0022] 일부 실시예들에서, OS(34)는 가상 메모리 공간(논리 어드레스 공간이라고도 함)을 구성하고, 상기 가상 메모리 공간을 도 3의 어플리케이션(42e-f 및 44)과 같은 어플리케이션에 그리고/또는 보호된 오브젝트(36a-b)와 같은 다른 소프트웨어 오브젝트에 노출시킨다. 이러한 시스템들에서, OS(34)는 예컨대 페이지 테이블 메커니즘을 사용하여 VM(32)의 가상화된 물리적 메모리와 상기 가상 메모리 공간 사이의 맵핑을 구성 및 유지시킨다. 일부 실시예들에서, 상기 가상 메모리 공간은 또한 페이지들로 구획되며, 이러한 페이지들은 OS(34)에 의해 가상화된 물리적 메모리로 개별적으로 맵핑된 가상 메모리의 최소 단위를 나타낸다(가상 메모리에서 가상화된 물리적 메모리로의 맵핑은 페이지 그래뉼러리티로 실행된다).

[0023] 도 4는 도 2에 도시된 바와 같은 실시예의 메모리 어드레스들의 예시적인 맵핑을 나타낸다. 어플리케이션(42a) 또는 게스트 OS(34a)와 같은 소프트웨어 오브젝트는 게스트 OS(34a)에 의해 가상 어드레스 공간(214a)을 할당받는다. 개별 소프트웨어 오브젝트가 예시적인 메모리 어드레스(50a)에 대한 액세스를 시도하면, 어드레스(50a)는 게스트 OS(34a)에 의해 구성 및 제어되는 페이지 테이블들에 따라 게스트 VM(32a)의 가상화된 프로세서에 의해 가상 머신(32a)의 가상화된 물리적 메모리 공간(114a) 내의 어드레스(50b)로 번역된다. 어드레스(50b)는 본 기술분야에서 게스트-물리적 어드레스라고도 알려진다. 그러면, 가상화된 물리적 메모리(114a)를 구성 및 제어하는 하이퍼바이저(30)는 예컨대 상술한 EPT 또는 NPT 수단을 사용하여, 어드레스(50b)를 호스트 시스템(10)의 물리적 메모리(14) 내의 어드레스(50c)로 맵핑한다.

[0024] 이와 유사하게, 가상 메모리 공간(214b)은 게스트 VM(32b) 상에서 실행되는 다른 소프트웨어 또는 어플리케이션들(예컨대, 42c)을 위해 게스트 OS(34b)에 의해 설정된다. 공간(214b) 내의 예시적인 가상 어드레스(50d)는, 게스트 OS(34b)에 의해 구성 및 제어되는 번역 테이블에 따라, 게스트 VM(32b)의 가상화된 프로세서에 의해 게스트 VM(32b)의 가상화된 물리적 메모리 공간(114b) 내의 어드레스(50e)로 번역된다. 어드레스(50e)는 하이퍼바이저(30)에 의해 물리적 메모리(14) 내의 어드레스(50f)로 또한 맵핑된다.

[0025] 일부 실시예들에서, 하이퍼바이저(30)는 물리적 메모리(14)의 리프레젠테이션을 포함하는 그 자신의 가상 메모리 공간(214c)을 설정하고, 공간(214c) 내의 어드레스를 물리적 메모리(14) 내의 어드레스로 맵핑하는 번역 메커니즘(예컨대, 페이지 테이블)을 사용한다. 도 4에서, 그러한 예시적인 맵핑은 어드레스(50g)를 어드레스(50h)로 번역한다. 이와 유사하게, 물리적 메모리(14) 내의 50c 및 50f와 같은 어드레스들은 하이퍼바이저(30)의 가상 메모리 공간(214c) 내에서 각각 어드레스들(50k 및 50m)에 대응한다.

[0026] OS(34a-b)의 본질적 임무는 개별 VM(32a-b) 상에서 실행되는 소프트웨어 오브젝트들에 메모리 섹션을 동적으로 할당하는 것, 및 개별 프로세스에 의해 더 이상 필요하지 않을 때 재사용을 위해 그러한 섹션을 자유롭게 하는 것이다. 이러한 메모리 할당은 가상 메모리(214a-b)의 레벨에서, 또는 개별 가상 머신의 레벨에서 실행되는 가상화된 물리적 메모리(114a-b)의 레벨에서 실행될 수 있다. 메모리 할당 및 반환은 통상적으로 Windows®의 KeAllocatePoolWithTag 및 KeFreePoolWithTag와 같은 전용 OS 메모리 관리 함수들에 의해 실행된다.

[0027] 전술한 메모리 맵핑 메커니즘으로 인해, 타겟 오브젝트에 대한 메모리 할당은 항상 타겟 오브젝트에 대한 개별 VM의 가상화된 물리적 메모리의 섹션의 할당을 유발한다. 개별 섹션의 크기는 개별 소프트웨어 오브젝트의 메모리 요구 및 페이지 크기에 따라, 페이지의 오직 일부분(fraction)이거나, 또는 한 페이지를 초과할 수 있다. 이하에서, 페이지 세트의 각 페이지가 개별 섹션의 일부를 포함할 경우, 섹션은 페이지 세트를 스팬(span)한다고 언급된다. 몇몇 오브젝트들은 동일 페이지의 구분되는 섹션을 할당받을 수 있다. 도 5는 복수의 페이지

(26a-d)로 분할된 가상화된 물리적 메모리의 예시적 영역 및 예시적인 메모리 할당을 도시하며, 구분되는 소프트웨어 오브젝트에 각각 할당된 2개의 섹션(52 및 54)은 동일한 페이지(26b)를 스팬한다. 또 다른 예시적 섹션(56)은 2개의 페이지(26c-d)를 스팬한다.

[0028] 이와 대조적으로, 도 6은 본 발명의 일부 실시예들에 따른 예시적인 메모리 할당을 도시한다. 이하에서 상세히 설명되는 바와 같이, 타겟 오브젝트가 멀웨어 보호를 필요로 할 때, 예컨대 타겟 오브젝트가 보호 영역(38)에 속하는 다른 오브젝트이거나 드라이버 오브젝트인 경우, 보호 프라이밍 모듈(46)의 일부 실시예들은 타겟 오브젝트에 대한 메모리 할당을 수정하여, 타겟 오브젝트의 일부를 포함하는 각 페이지가 타겟 오브젝트에 대해 예약되게 한다.

[0029] 통상의 기술자는 타겟 오브젝트에 대한 페이지 세트의 예약이 여러 방법으로 달성될 수 있다는 것을 이해할 것이다. 일부 실시예들에서, 페이지를 예약하는 것은 도 6의 예시에 도시된 바와 같이, 타겟 오브젝트에 전체 페이지를 할당하는 것을 포함하며, 섹션(58)은 전체 페이지(26f)를 점유한다. 이러한 독점적 할당은, 예컨대 이하에서 더욱 상세히 설명하는 바와 같이, 개별 오브젝트에 할당된 섹션의 크기를 페이지 크기의 정수배와 동일한 새로운 크기로 변경하는 것에 의해 달성될 수 있다. 이와 대조적으로, 도 6의 동일한 예시에서, 미보호 소프트웨어 오브젝트는 다른 페이지(26g)의 단지 일부분으로 구성된 섹션(60)을 할당받는다. 다른 실시예들에서, 타겟 오브젝트에 대해 페이지 세트를 예약하는 것은 타겟 오브젝트의 크기를 변경하는 것 또는 타겟 오브젝트에 페이지 세트의 전체를 할당하는 것을 반드시 포함할 필요는 없지만, 그렇지 않다면 타겟 오브젝트 이외의 어떠한 오브젝트도 개별 페이지 세트 내의 할당된 메모리 공간을 할당받지 않도록 보장함으로써 달성될 수 있다. 예를 들어, 타겟 오브젝트에 대해 페이지를 예약하는 것은, 이하에서 더욱 상세히 설명하는 바와 같이, 예컨대 OS(34)의 초기화시에 빈 페이지 세트를 제외하는 것, 및 오브젝트를 빈 페이지의 개별 세트 내로 할당하는 것(또는 이동하는 것)을 포함할 수 있다.

[0030] 도 7은 본 발명의 일부 실시예들에 따른 메모리 인트로스펙션 엔진(40)에 의해 실행되는 단계들의 예시적 시퀀스를 도시한다. 엔진(40)은 호스트 시스템(10) 상에서 동시에 실행되는 복수의 가상 머신을 검사 및/또는 보호하도록 구성될 수 있다. 도 7에 도시된 단계들은 이러한 가상 머신 각각에 대해 반복될 수 있다. 단계(102)에서, 엔진(40)은 OS(34)의 초기화를 검출한다. 일부 실시예들에서, 단계(102)는 하이퍼바이저(30)의 레벨로부터 OS 초기화를 나타내는 포로세서 이벤트들을 청취하는 단계를 포함한다. OS 초기화 동작을 수행하는 프로세서 명령들은 통상적으로 커널 프로세서 권한, 예컨대 Intel 플랫폼에서의 링 0를 필요로 한다. OS(34)의 권한 레벨로부터 실행될 때, 이러한 명령들은 프로세서(12)의 제어권을 OS(34)로부터 하이퍼바이저(30)로 넘기는, Intel 플랫폼에서의 VMExit와 같은 가상 머신 엑시트 이벤트를 촉발할 수 있다. 그에 따라, 가상 머신 엑시트 이벤트들은 하이퍼바이저(30) 및/또는 인트로스펙션 엔진(40)에 의해 분석될 수 있다. OS 초기화를 나타내는 프로세서 명령 및/또는 이벤트들의 예는 LIDT 및/또는 LGDT와 같은 권한 명령들(previlieged instructions)을 사용하여, Windows OS의 초기화시에 일찍 이루어지는, 인터럽트 및 글로벌 디스크립터 테이블의 초기화를 포함한다. OS 초기화를 나타내는 이벤트들/명령들의 다른 예는 OS의 커널 이미지가 메모리에 로딩된 후 즉시 이루어지는 WRMSR 기계 명령을 사용하여, SYSENTER 및/또는 SYSCALL 모델-특이적 레지스터들에 기록하는 것을 포함한다. 또 다른 예들은 특히 기계-체크 구성 레지스터, 및 FS/GS 베이스 레지스터와 같은 OS(34)에 의해 프로그래밍된 다른 모델-특이적 레지스터들을 포함한다. 일부 실시예들에서, 이러한 이벤트들/명령들을 검출하는 단계는 OS(34)가 초기화되었음을 나타낸다.

[0031] 단계(104)에서, 메모리 인트로스펙션 엔진(40)의 일부 실시예들은 개별 게스트 VM 상에서 현재 실행중이거나 초기화된 운영 시스템의 유형을 식별한다. 예시적인 OS 유형은 특히 Windows, Linux, MacOS, 및 Android를 포함한다. OS 유형은 특히 Windows와 같은 명칭 인디케이터, 7, 가정용, 또는 기업용과 같은 버전 인디케이터를 포함할 수 있다. 일부 실시예들에서, 단계(104)는 개별 게스트 VM의 모델-특이적 레지스터(MSR)의 콘텐츠에 따른 또는 개별 MSR에 의해 지시된 메모리 섹션의 콘텐츠에 따라 OS의 유형을 식별하는 단계를 포함한다. 일부 실시예들에서, 엔진(40)은 단계(102)에서 인터셉트된 명령에 의해 그러한 MSR에 기록된 데이터에 따라 OS의 명칭을 결정할 수 있다. 예를 들어, 엔진(40)은 SYSENTER 또는 SYSCALL MSR에 대한 쓰기 명령을 인터셉트하여, 이러한 쓰기 명령의 파라미터에 따라 현재 실행 중이거나 현재 초기화 중인 OS의 유형을 결정할 수 있다. OS 명칭에 관한 정보를 제공할 수 있는 다른 예시적인 레지스터들은 특히 제어 레지스터, 인터럽트 디스크립터 테이블(IDT), 및 글로벌 디스크립터 테이블(GDT)을 포함한다. MSR 쓰기에 따라 OS를 식별하기 위해, 인트로스펙션 엔진(40)은 각각의 OS에 대해 특이적인 패스트 시스템-콜 핸들러들(fast system-call handlers)(예컨대, SYSCALL 또는 SYSENTER MSR의 콘텐츠에 따라 취급되는 시스템 콜)의 미리 결정된 라이브러리에 대한 패턴 매칭을 더 사용할 수 있다. 이러한 신속 시스템-콜 라이브러리에는 메모리 인트로스펙션 엔진(40)이 제공될 수 있으며, 주

기적인 또는 요청에 따른(on-demand) 소프트웨어 업데이트를 통해 최신으로 유지될 수 있다.

[0032] 일부 실시예들에서, 버전 인디케이터(릴리즈 명칭, 빌드 넘버 등과 같은)는 OS의 개별 유형에 대해 특이적인 특정 커널 데이터 구조를 파싱(parsing)함으로써 얻어질 수 있다. OS 버전의 식별을 허용하는 예시적은 데이터 구조는 특히 NtBuildNumber와 같은 Windows 커널의 특정 내보내기(export) 심볼 또는 Linux 커널의 특정 내보내기 심볼이다.

[0033] 단계(106)에서, 메모리 인트로스펙션 엔진(40)은 엔진(40)에 대한 일체성 보호의 적용을 준비하는 과정에서, 보호 프라이밍 모듈(46)에게 메모리-프라이밍 동작의 수행을 요청할 수 있다. 일부 실시예들에서, 이러한 메모리 프라이밍 동작은 멀웨어 보호를 위해 선택된 타겟 오브젝트의 할당을 수정하는 단계를 포함함으로써, 타겟 오브젝트의 부분들을 포함하는 모든 페이지가 개별 타겟 오브젝트를 위해 예약되게 한다. 일부 실시예들에서, 단계(106)는 멀웨어 보호를 위해 선택된 오브젝트들에게 추후 독점적으로 할당되도록, 미할당 메모리 페이지들의 예약된 풀(reserved pool)을 수립하는 단계(establishing)를 포함할 수 있다. 이러한 예약된 풀을 수립하기 위해, 보호 프라이밍 모듈(46)은 미리결정된 크기(예컨대, 20MB)의 더미 소프트웨어 오브젝트에 개별 풀을 할당하기 위해 네이티브 OS 메모리 할당을 요청할 수 있다. 따라서, 모듈(46)은 네이티브 OS 메모리 관리 기능에 의해 수행되는 추가적인 할당시 메모리의 개별 풀이 미사용될 것을 보장할 수 있다. 모듈(46)의 이러한 기능은 도 8-10과 관련하여, 이하에서 더욱 상세히 설명된다.

[0034] 단계(108)에서, 메모리 인트로스펙션 엔진(40)은 보호를 위한 타겟 오브젝트를 선택한다. 일부 실시예들에서, 단계(108)는 단계들(102-104)에서 결정된 OS의 유형에 따라, 멀웨어 보호[도 3의 보호 영역(38) 참조]가 요구되는 소프트웨어 오브젝트를 식별하는 단계를 포함한다. 보호를 위한 대상이 되는 예시적인 오브젝트들은 특히 OS 드라이버 오브젝트들 및 안티-멀웨어 드라이버(36b)이다. 이러한 오브젝트들을 식별하기 위해, 메모리 인트로스펙션 엔진(40)은 멀웨어 보호를 요구하는 OS-특이적 오브젝트들의 리스트를 유지할 수 있으며, 상기 리스트는 주기적인 및/또는 요청에 따른 소프트웨어 업데이트에 의해 최신으로 유지될 수 있다. 일부 실시예들에서, 엔진(40)은 OS(34)에 의한 드라이버 로드의 시도를 인터셉트하여, 개별 드라이버의 서명 및/또는 일체성 세트의 체크를 실행할 수 있다. 이러한 체크는 알려진 드라이버들에 대해 결정되어 있는 해시(hash)의 라이브러리에 대해 개별 드라이버의 일부의 해시를 매칭하는 단계를 포함할 수 있으며, 로드를 위해 OS(34)가 사용중인 그리고/또는 현재 시도중인 복수의 드라이버 중에서 안티-멀웨어 드라이버(36b)를 식별하게 할 수 있다. 식별이 이루어지면, 드라이버(36b)는 이하에서 더욱 상세히 설명하는 바와 같이, OS의 구성요소와 더불어 보호될 수 있다.

[0035] 단계(110)는 타겟 오브젝트가 초기화될 때까지 대기한다. 예를 들면, 드라이버가 초기화될 때, OS(34)는 개별 드라이버에 할당된 메모리 공간에 대한, 또는 개별 드라이버에 따라, 또는 개별 드라이버가 제어하도록 구성된 장치에 따라 선택된 다른 소프트웨어 오브젝트에 대한, 다수의 적법한(legitimate) 구성 기록을 실행하고, 엔진(40)은 이러한 적법한 기록이 단계(110)의 일부로써 진행되게 할 수 있다. 타겟 오브젝트가 초기화되었는지를 결정하기 위해, 엔진(40)의 일부 실시예들은 드라이버 초기화를 나타내는 이벤트 및/또는 프로세서 명령을 청취할 수 있다. 이러한 이벤트의 예에는 특히 개별 오브젝트의 특정 영역에 대한 수정이 포함된다.

[0036] 단계(112)에서, 엔진(40)은 예컨대 OS(34)와의 타협을 시도하는 악의적 소프트웨어에 의한 원치않는 수정으로부터 개별 타겟 오브젝트를 보호한다. 이러한 몇몇 메모리 보호 메커니즘이 본 기술분야에 알려져 있다. 일부 실시예들에서, 타겟 오브젝트를 보호하는 단계는 개별 타겟 오브젝트에 대해 OS(34)에 의해 할당된 메모리 공간을 쓰기-방지(write-protect)하는 단계를 포함한다. 이러한 쓰기-방지는 EPT 또는 NPT와 같은 데이터 구조를 사용하여 메모리 인트로스펙션 엔진(40)에 대한 요청에 따라 하이퍼바이저(30)에 의해 강제될 수 있다. 예를 들어, 하이퍼바이저(30)는 개별 페이지의 EPT/NPT 액세스 권한 비트를 수정함으로써, 타겟 오브젝트에 할당된 메모리 페이지를 리드-온리(read-only)로 설정할 수 있다. 대안적으로, 하이퍼바이저(30)는 타겟 오브젝트에 할당된 메모리 페이지에 대한 임의의 쓰기 시도를 인터셉트하여, 분석을 위해 그러한 개별 시도를 메모리 인트로스펙션 엔진(40)으로 재지향시킬 수 있다. 쓰기 시도를 분석한 후, 메모리 인트로스펙션 엔진(40)은 개별 쓰기 동작을 허용할지, 아니면 거부(드롭)할지를 결정할 수 있다. 단계들(108-112)은 멀웨어 보호를 필요로 하는 모든 타겟 오브젝트들에 대해 반복될 수 있다.

[0037] 일부 실시예들에서, 보호 프라이밍 모듈(46)에 의해 실행되는 메모리 프라이밍 동작은 멀웨어 보호를 위해 선택된 타겟 소프트웨어 오브젝트의 메모리 할당을 수정하는 단계를 포함하며, 상기 수정 단계는 앞서 설명한 바와 같은[도 7, 단계(108-112)], 개별 소프트웨어 오브젝트의 일체성 보호를 적용하는 엔진(40)에 선행하여 실시된다. 일부 실시예들에서, 모듈(46)은 메모리 할당을 수정하여, 타겟 오브젝트의 일부를 포함하는 메모리 페이지

가 개별 타겟 오브젝트에 대해 예약되게 한다. 상기 메모리 할당의 수정 단계는 OS(34)에 대해 네이티브한 메모리 관리 함수를 실행한 결과를 수정하는 단계, 및/또는 개별 메모리 관리 함수 그 자체를 수정하는 단계를 포함할 수 있다. 네이티브 메모리 관리 함수들은 OS(34) 제조자에 의해 제공되는 소프트웨어 오브젝트들을 포함하고, 네이티브 함수들은 메모리 할당 및 반환 동작을 실행한다. 이러한 함수들의 예로는 Windows OS에 대해 네이티브인 KeAllocatePoolWithTag 및 KeFreePoolWithTag가 있다.

[0038] 일부 실시예들에서, 보호 프라이밍 모듈(46)은, 예컨대 그러한 함수들이 개별 게스트 가상 머신(32)의 메모리에 속해 있는 메모리 어드레스들을 결정함으로써, 단계(104)에서 인트로스펙션 엔진에 의해 결정된 OS 유형에 따라 네이티브 메모리 관리 함수들의 세트를 식별한다. 이러한 메모리 어드레스들을 결정하기 위해, 모듈(46)은 커널 바이너리 이미지(예컨대, Windows의 Portable Executable, Linux의 Executable 및 Linkable Format)의 내보내진 함수 테이블과 같은 특정 데이터 구조를 액세스할 수 있다.

[0039] 네이티브 메모리 관리 함수들의 식별에 이어, 모듈(46)의 일부 실시예들은 추가적인 기능을 제공함으로써 상기 함수들의 수정을 진행할 수 있다. 이러한 수정은 본 기술분야에서 알려진 임의의 훅킹(hooking) 방법을 사용하여 달성될 수 있다. 예를 들어, 모듈(46)은 개별 네이티브 함수에 중복쓰기 되거나 이에 부가된 VMCall 명령 또는 JMP 명령과 같은 재지향 패치(re-direction patch)를 적용할 수 있다. 다른 실시예들은 새로운 어드레스를 지향하도록, 개별 메모리 관리 함수의 EPT 엔트리를 수정할 수 있다. 일부 실시예들에서, 이러한 패치 및/또는 EPT 후크의 효과는 보호 프라이밍 모듈(46)에 의해 제공된 코드의 일부분으로 네이티브 함수의 실행을 재지향하는 것이며, 이러한 코드의 예시적인 기능이 이하에서 제공된다. 훅킹에 이어, OS(34)가 타겟 오브젝트에 대한 메모리 할당 또는 타겟 오브젝트로부터의 메모리 반환을 시도할 때, 코드의 일부분은 개별 네이티브 OS 메모리 관리 함수의 코드 이전에 또는 그를 대신하여 실행될 것이다.

[0040] 도 8은 본 발명의 일부 실시예들에 따른, Windows의 KeAllocatePoolWithTag와 같은 네이티브 OS 메모리 할당 함수를 수정하는 단계를 포함하는 단계들의 예시적인 시퀀스를 도시한다. 상기 시퀀스는 보호 프라이밍 모듈(46)의 일부를 구성할 수 있고, 개별 메모리 할당 함수에 적용되는 패치/후크에 의해 실행되는 재지향의 결과로써 실행될 수 있다. 단계(122)는 메모리 할당을 요청하는 오브젝트가 멀웨어 보호의 자격이 있는지를 결정하기 위한 선택 기준을 적용한다. 이러한 결정은 예를 들어 개별 함수 콜(call)의 파라미터 및/또는 아규먼트에 따라 실행될 수 있다. 예시적인 선택 기준은 오브젝트의 유형, 예컨대 드라이버 오브젝트에 따라 오브젝트를 선택하는 단계를 포함한다. Windows 패밀리로부터 OS(34)를 구동하는 실시예에서, 할당되는 오브젝트의 유형은 네이티브 메모리 할당 함수 KeAllocatePoolWithTag의 할당 태그에 따라 결정될 수 있다. 예를 들어, 'Driv' 태그는 드라이버 오브젝트를 나타낸다. 대안적인 선택 기준은 할당을 요청하는 오브젝트가 멀웨어 보호를 목적으로 하는 오브젝트의 리스트에 있는지를 결정하는 단계를 포함한다. 이러한 리스트는 보호 영역(38)(도 3)의 멤버들을 포함할 수 있다. 일부 실시예들에서, 추가적인 선택 기준은 개별 오브젝트의 메모리 할당이 전술한 방법에 의해 안전하게 수정되었는지를 결정하는 단계를 포함한다.

[0041] 단계(124)는 타겟 오브젝트가 멀웨어 보호를 위한 선택 기준을 만족하는지를 결정한다. 만일 아니라면, 단계(128)는 개별 OS의 네이티브 메모리 할당 함수로 실행 제어권을 넘겨 준다. 타겟 오브젝트가 보호를 위해 선택되면, 단계(126)에서, 보호 프라이밍 모듈(46)의 일부 실시예들은 페이지 크기의 정수배와 동일한 새로운 크기로 타겟 오브젝트의 크기를 변경한다. 이러한 방식으로 오브젝트의 크기를 변경하는 것은 오브젝트를 수용하는 데 필요한 메모리의 실제 양 대신에, 개별 타겟 오브젝트의 전체 페이지 세트를 할당하도록 메모리 할당자(memory allocator)를 효율적으로 강제할 수 있다. 일부 실시예들에서, 타겟 오브젝트의 크기는 다음으로 가장 가까운 정수배의 페이지 크기로 라운딩된다. 예를 들어, 320 byte의 타겟 오브젝트는 전체 4kB 페이지를 할당받을 수 있고, 6kB의 오브젝트는 2개의 전체 4kB 페이지를 할당받을 수 있는 것 등이다. 일부 실시예들, 예컨대 Windows OS를 실행하는 가상 머신에서, 크기의 면에서 적어도 한 페이지를 측정하는 메모리 섹션을 할당하는 것에 의해, 할당된 섹션이 페이지 바운더리에 자동으로 정렬된다[예컨대, 도 6의 섹션(58)]. 다른 실시예들에서, 단계(126)는, 섹션에 의해 스팬된 모든 페이지들이 타겟 오브젝트에 독점적으로 할당되도록, 할당된 섹션을 페이지 바운더리에 명확하게 정렬시키는 단계를 포함할 수 있다. 통상의 기술자는 페이지 바운더리에 대한 섹션의 정렬을 달성하기 위한 많은 방법이 있다는 것을 이해할 것이다. 예를 들어, 단계(126)는 한 페이지의 크기만큼 타겟 오브젝트의 크기를 증가시키는 단계, 및 포인터를 결과적인 메모리 할당 섹션으로 변경하는 단계를 포함할 수 있다. 단계(126)가 완료된 후, 실행 제어권은 네이티브 OS 메모리 관리 함수로 넘겨진다.

[0042] 도 9는 보호 프라이밍 모듈(46)이 메모리 할당을 달성할 수 있는 대안적인 방법을 설명하는 단계들의 예시적인 시퀀스를 도시하며, 보호되는 소프트웨어 오브젝트에 할당된 섹션에 의해 스팬된 모든 페이지들은 개별 오브젝트를 위해 예약되어 있다. 도 9에 도시된 시퀀스는 Windows의 KeAllocatePoolWithTag와 같은 네이티브 OS 메

모리 할당 함수의 기능을 수정하는 단계를 포함하고, 개별 메모리 할당 함수에 대해 적용된 패치/후크에 응답하여 실행될 수 있다. 단계들(132-134)의 시퀀스는 타겟 오브젝트가 멀웨어 보호를 위한 기준을 충족하는지를 확인하고, 단계들(132-134)은 전술한 단계들(122-124)과 유사한 방식으로 진행될 수 있다. 타겟 오브젝트가 보호를 위해 선택되지 않으면, 단계(138)는 프로세서의 제어권을 개별 네이티브 메모리 할당 함수로 넘겨 준다. 타겟 오브젝트가 멀웨어 보호를 위해 선택되면, 단계(136)는 네이티브 할당 함수를 우회하고, 보호되는 오브젝트를 위해 예약된 메모리 영역 내에 위치한 섹션을 직접 할당한다. 일부 실시예들에서, 단계(136)는 OS(34)의 초기화시 모듈(46)에 의해 수립된 예약된 메모리 풀 내에 위치한 어드레스를 나타내는 할당 포인터를 결정하는 단계를 포함한다[도 7, 단계(106) 관련 사항 참조]. 일부 실시예들에서, 모듈(46)은 할당 포인터를 또한 결정함으로써, 할당된 섹션을 페이지 바운더리에 대해 정렬되게 할 수 있다. 다음으로, 단계(140)는 단계(136)에서 결정된 할당 포인터를 복귀시킨다.

[0043] 일부 실시예들에서, 보호 프라이밍 모듈(46)은 네이티브 OS 메모리 반환 함수의 실행 결과를 변경함으로써, 또는 개별 함수 그 자체를 수정함으로써, 반환 프로세스를 또한 수정할 수 있다. 이러한 수정은 예컨대 추가적인 기능을 포함하도록 KeFreePoolWithTag와 같은 네이티브 OS 메모리 반환 함수를 훅킹함으로써 달성될 수 있다. 도 10은 본 발명의 일부 실시예들에 따른, 이러한 메모리 반환 함수의 수정을 포함하는 단계들의 예시적인 시퀀스를 도시한다. 단계들(142-144)의 시퀀스는 반환되는 오브젝트의 어드레스가 인트로스펙션 엔진(40) 및/또는 하이퍼바이저(30)에 의해 쓰기-방지되어 있는지를 결정한다(위 참조). 개별 어드레스/페이지가 이러한 보호상태가 아니면, 단계(148)는 실행 제어권을 네이티브 OS 반환 함수로 넘겨 준다. 개별 어드레스/페이지가 엔진(40) 및/또는 하이퍼바이저(30)에 의해 쓰기-방지되어 있다면, 단계(146)에서, 메모리 인트로스펙션 엔진(40) 및/또는 하이퍼바이저(30)의 구성요소는 단계(148)로 진행하기 전에 그러한 보호를 제거할 수 있다. 일부 실시예들에서, 개별 오브젝트로부터 보호를 제거하는 단계는 개별 오브젝트의 어드레스를 수용하는 페이지가 더이상 쓰기-방지, 예컨대 리드-온리가 아님을 나타내도록, EPT 또는 NPT 내의 변경을 연산하는 단계를 포함한다.

[0044] 전술한 예시적인 시스템 및 방법은 바이러스 및 루트킷과 같은 멀웨어로부터 컴퓨터 시스템과 같은 호스트 시스템의 보호를 허용한다. 종래 시스템의 경우, 루트킷은 운영 시스템의 권한 레벨과 실질적으로 유사한 프로세서 권한 레벨에서 작동하여 컴퓨터 시스템의 제어권을 취할 수 있었다. 이와 대조적으로, 본 발명의 일부 실시예들에서는 하이퍼바이저가 최고 권한 레벨로 컴퓨터 시스템상에서 실행됨으로써, 운영 시스템을 가상 머신으로 바꿀 수 있다. 일부 실시예들에서, 메모리 인트로스펙션 엔진은 하이퍼바이저와 동일한 프로세서 권한 레벨로 실행된다. 그에 따라, 안티-멀웨어 작동은 운영 시스템의 권한 레벨보다 높은 프로세서 권한 레벨로부터 실행될 수 있다. 일부 실시예들에서, 단일 메모리 인트로스펙션 엔진은 개별 컴퓨터 시스템상에서 동시에 실행되는 다수의 가상 머신을 보호할 수 있다.

[0045] 일부 실시예들에서, 멀웨어로부터 개별 시스템을 보호하는 단계는 특히 특정 드라이버, 레지스터, 및 페이지 테이블과 같은 중요 소프트웨어 오브젝트의 세트를 선택하는 단계, 및 이러한 오브젝트에 대한 악의적인 변경을 방지하는 단계를 포함한다. 타겟 오브젝트를 보호하기 위해, 일부 실시예들은 개별 오브젝트에 할당된 메모리 공간에 대한 쓰기 시도를 인터셉트함으로써 이러한 악의적인 변경을 방지할 수 있다. 이러한 인터셉션은 하이퍼바이저의 레벨에서 실행될 수 있다.

[0046] 다른 실시예들은 개별 오브젝트에 할당된 메모리 공간을 리드-온리로 표시함으로써 타겟 오브젝트를 보호할 수 있다. 통상적인 하드웨어 및 소프트웨어 구성에서, 메모리는 페이지로 알려진 인접 어드레스들의 개별 블록으로 구획된다. 액세스 허가, 예컨대, 읽기 및 쓰기 허가는 통상적으로 페이지 그래뉼러리티로, 즉 한 페이지 내의 모든 어드레스들이 동일한 액세스 허가를 갖도록, 설정된다. 따라서, 타겟 오브젝트의 메모리 공간을 보호하는 단계는, 예를 들어 개별 오브젝트에 속하는 데이터를 포함하는 페이지 세트를 리드-온리로 표시함으로써 달성될 수 있다. 페이지-레벨 액세스 허가는 예컨대, Intel 플랫폼에서의 익스텐디드 페이지 테이블(EPT)과 같은 전용 데이터 구조를 사용하여, 하이퍼바이저에 의해 제어된다.

[0047] 종래 시스템의 경우, OS가 개별 시스템상에서 실행되는 소프트웨어 오브젝트에 메모리 공간을 할당할 때, 예컨대 개별 오브젝트가 작은 메모리 풋프린트를 가질 때에는, 다수의 소프트웨어 오브젝트가 동일한 페이지 내에 할당될 수 있었다. 몇몇 상황에서, 개별 오브젝트들 중 하나는 OS에게 있어 중요한 것이어서 멀웨어로부터의 보호를 필요로 할 수 있지만, 동일 페이지에서 다른 오브젝트는 빈번한 그리고 적법한 재쓰기(re-writing)를 필요로 할 수 있다. 액세스 허가가 페이지 그래뉼러리티로 설정되기 때문에, 중요 오브젝트를 보호하는 것에 의해 중요 오브젝트와 동일한 페이지 내에 있는 모든 오브젝트들에 대한 쓰기 액세스를 거부하는 것이 초래될 수 있다. 보호된 페이지 내에 있는 메모리 어드레스에 대한 각각의 쓰기 시도는, 통상적으로 오류(fault)를 초래하고, 그에 이어 프로세서의 제어권이 개별 가상 머신의 OS로부터 하이퍼바이저로 넘어가는 가상 머신 엑시트

이벤트가 뒤따른다. 이러한 이벤트는 연산 고정비에 더욱 추가되는, 프로세서로의 또는 프로세서로부터의 개별 가상 머신의 상태를 로딩 및/또는 언로딩하는 것을 포함할 수 있다. 따라서, 무엇보다도, 보호되는 오브젝트를 수용하는 페이지에 대한 적법한 쓰기 시도조차도, 컴퓨터 시스템의 상당한 속도저하를 유발할 수 있다.

[0048] 이와 대조적으로, 본 발명의 일부 실시예들은 멀웨어 보호를 필요로 하는 소프트웨어 오브젝트에 메모리가 할당되는 방식을 수정함으로써, 개별 오브젝트의 부분을 포함하는 각각의 메모리 페이지가 개별 오브젝트를 위해 예약되게 한다. 이러한 수정은 개별 소프트웨어 오브젝트에 대한 일체성 보호를 적용하기 전에 실행된다. 다른 임의의 오브젝트가 타겟 오브젝트와 메모리 페이지를 공유하지 않도록 보장함으로써, 본 발명의 일부 실시예들은 페이지 그래뉼러리티로 일체성 보호를 허용하면서도, 적법한 쓰기 시도의 인터셉트에 의해 유발되는 연산 고정비를 방지할 수 있다.

[0049] 일부 실시예들에서, 개별 OS에 대해 네이티브한 할당 방식의 수정 단계는 OS의 메모리 할당 및 반환 함수들을 식별하는 단계, 및 하이퍼바이저의 프로세서 권한 레벨에서 실행되는 명령 세트로 그들의 실행을 재지향하기 위해 개별 함수들을 훅킹하는 단계를 포함한다. 대안적으로, 상기 수정은 개별 함수들을 개별 OS를 구동하는 VM 내부에서 실행되는 명령 세트로 재지향시키며, 명령 세트는 하이퍼바이저의 레벨에서 개별 VM의 메모리로 주입된다. 일부 실시예들에서, 메모리 할당 함수는 보호가 필요한 소프트웨어 오브젝트에 대한 전체 메모리 페이지들의 할당을 강제하도록 수정되고, 할당된 섹션은 페이지 바운더리와 정렬된다. 따라서, 보호가 필요한 오브젝트는 보호가 필요하지 않은 오브젝트와 더이상 동일한 페이지에 할당되지 않을 수 있다.

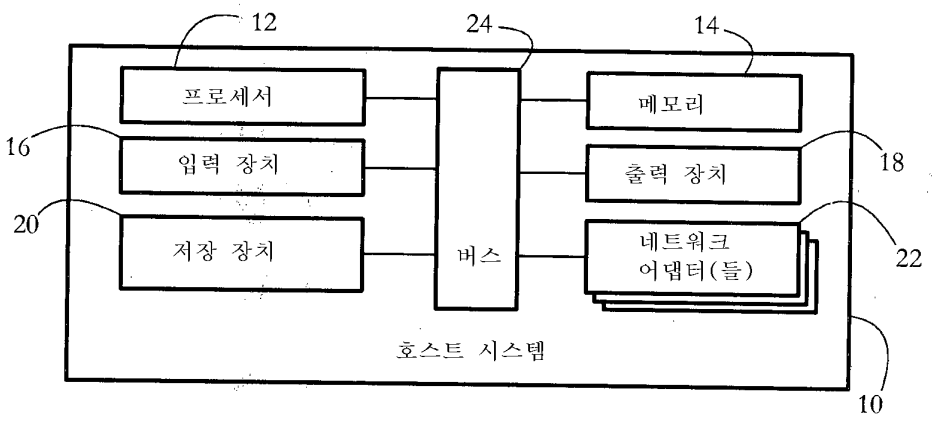
[0050] 대안적인 실시예는, 메모리의 예약된 풀을 비워둠으로써, 예컨대 OS 초기화시에, 타겟 오브젝트에 할당된 각각의 페이지가 타겟 오브젝트에 대해 예약되도록 보장한다. 운영 시스템이 타겟 오브젝트를 위한 메모리 할당을 요청할 때, 일부 실시예들은 할당 포인터를 예약된 풀 내의 어드레스로 재지향할 수 있고, 이로 인해 타겟 오브젝트를 예약된 페이지 세트로 효율적으로 이동시킬 수 있다.

[0051] 종래의 안티-멀웨어 솔루션들은 통상적으로 단일 운영 시스템에 대해 설계되었다. 하나의 운영 시스템과 다른 운영 시스템 사이의 스위칭은 상이한 버전의 안티-멀웨어 소프트웨어의 로딩을 필요로 할 수 있다. 이에 반해, 본 발명의 일부 실시예들의 경우에는 동일한 메모리 인트로스펙션 엔진이 현재 구동중인 운영 시스템의 유형 및 버전에 관계없이, 개별 컴퓨터 시스템의 멀웨어-보호를 할 수 있다. 하이퍼바이저의 레벨에서 메모리 인트로스펙션 엔진을 실행함으로써, 그리고 운영 시스템을 가상 머신으로 바꿈으로써, 일부 실시예들은 여러 운영 시스템을 동시에 구동 및 보호할 수 있다. 일부 실시예들에서, 메모리 인트로스펙션 엔진은 예컨대 부트-업(boot-up) 시에 각각의 운영 시스템을 동적으로 식별할 수 있고, OS-특이적 소프트웨어 오브젝트 및 데이터 구조를 또한 보호할 수 있다.

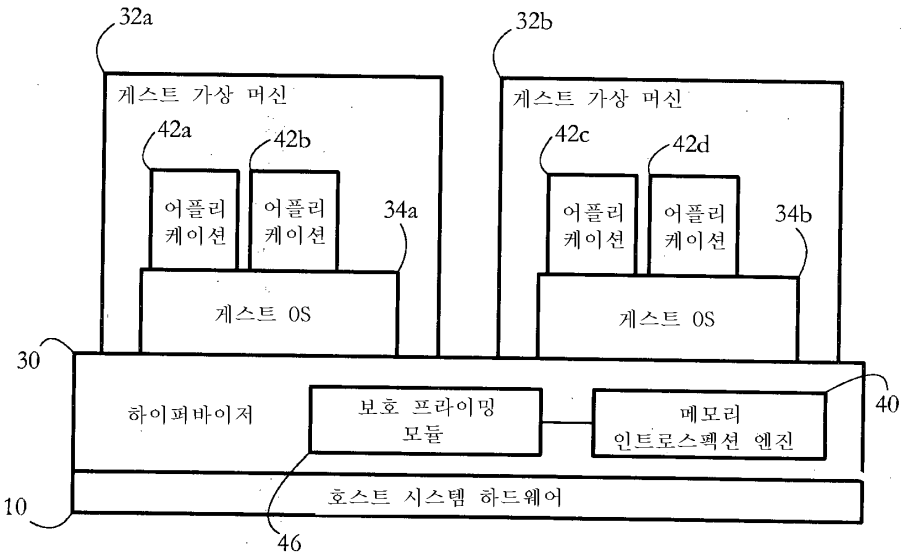
[0052] 본 발명의 범위를 벗어나지 않으면서 상기 실시예들이 많은 방식으로 변형될 수 있다는 것은 통상의 기술자에게 명확할 것이다. 따라서, 본 발명의 범위는 이어지는 특허청구범위 및 그들의 법적인 등가물에 의해 결정되어야만 한다.

도면

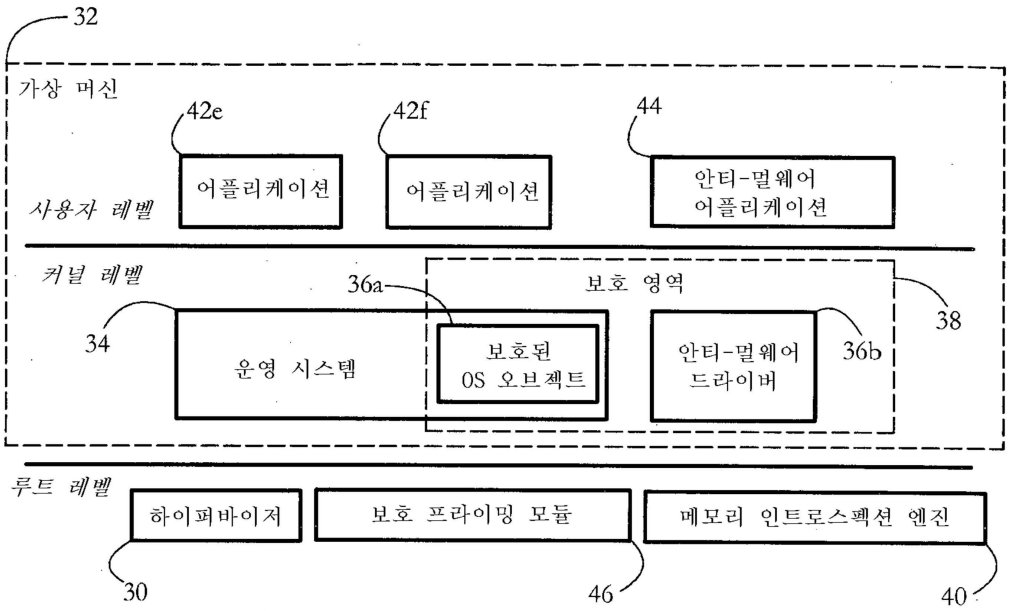
도면1



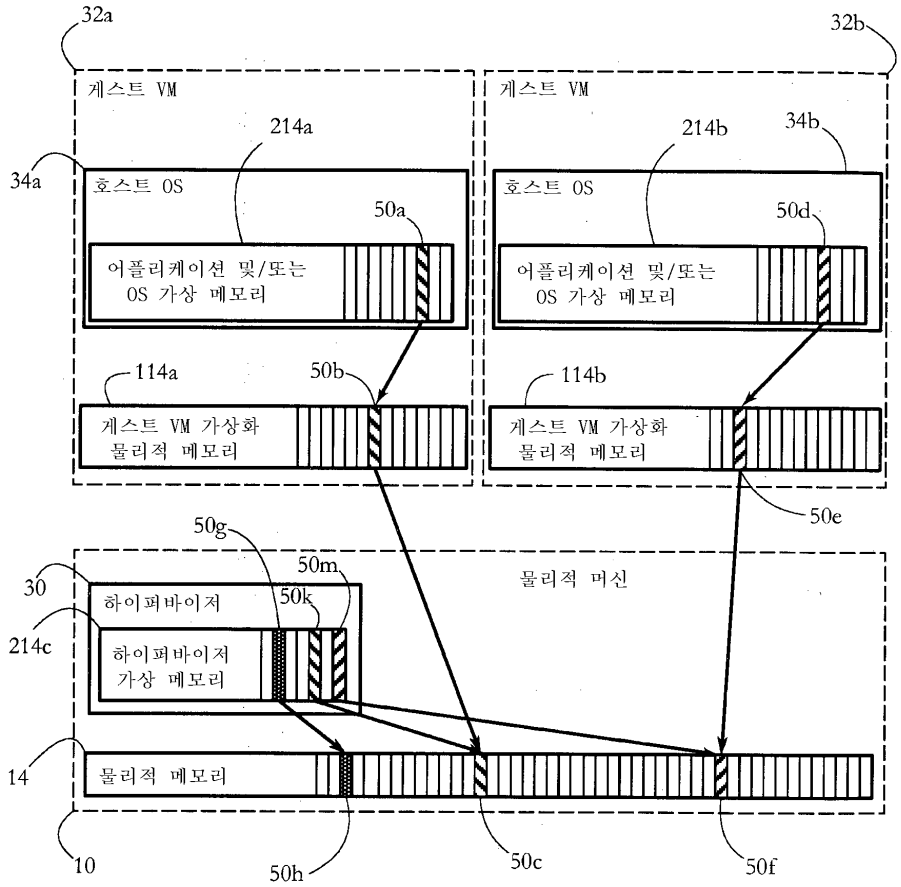
도면2


32a
32b
게스트 가상 머신
게스트 가상 머신
42a
42b
42c
42d
어플리케이션
어플리케이션
어플리케이션
어플리케이션
34a
34b
게스트 OS
게스트 OS
30
하이퍼바이저
보호 프라이밍 모듈
메모리 인트로스펙션 엔진
40
10
호스트 시스템 하드웨어
46

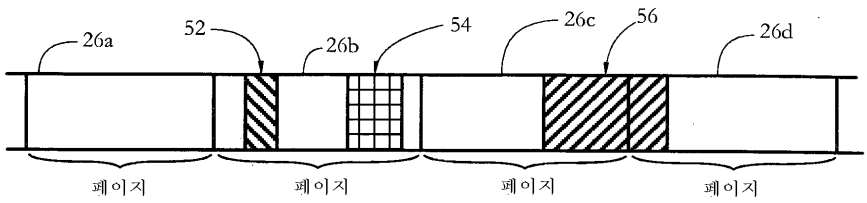
도면3


32
가상 머신
42e
42f
44
사용자 레벨
어플리케이션
어플리케이션
안티-멀웨어 어플리케이션
커널 레벨
36a
보호 영역
38
34
운영 시스템
보호된 OS 오브젝트
안티-멀웨어 드라이버
36b
루트 레벨
하이퍼바이저
보호 프라이밍 모듈
메모리 인트로스펙션 엔진
30
46
40

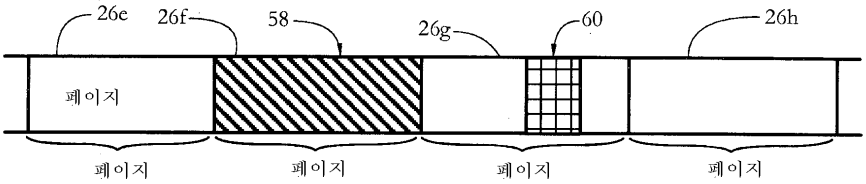
도면4


32a
32b
게스트 VM
게스트 VM
214a
214b
34b
34a
호스트 OS
호스트 OS
50a
50d
어플리케이션 및/또는
OS 가상 메모리
어플리케이션 및/또는
OS 가상 메모리
114a
50b
114b
게스트 VM 가상화
물리적 메모리
게스트 VM 가상화
물리적 메모리
50e
50g
50m
물리적 머신
30
하이퍼바이저
50k
214c
하이퍼바이저
가상 메모리
14
물리적 메모리
10
50h
50c
50f

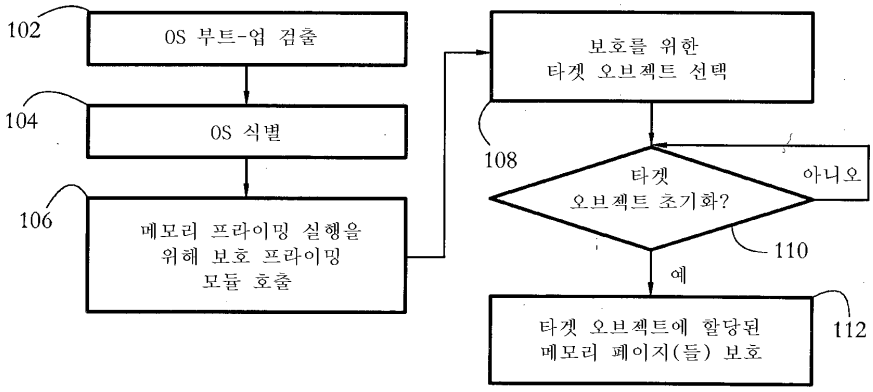
도면5


26a
52
26b
54
26c
56
26d
페이지
페이지
페이지
페이지

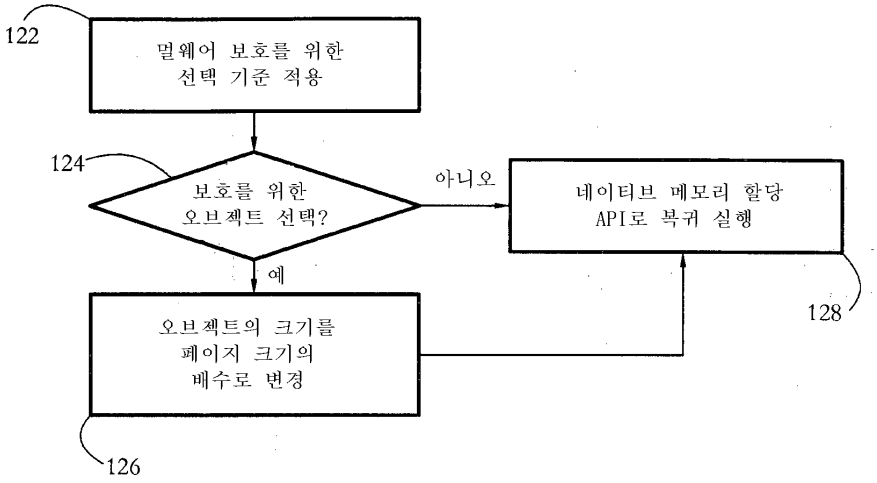
도면6


26e
26f
58
26g
60
26h
페이지
페이지
페이지
페이지
페이지

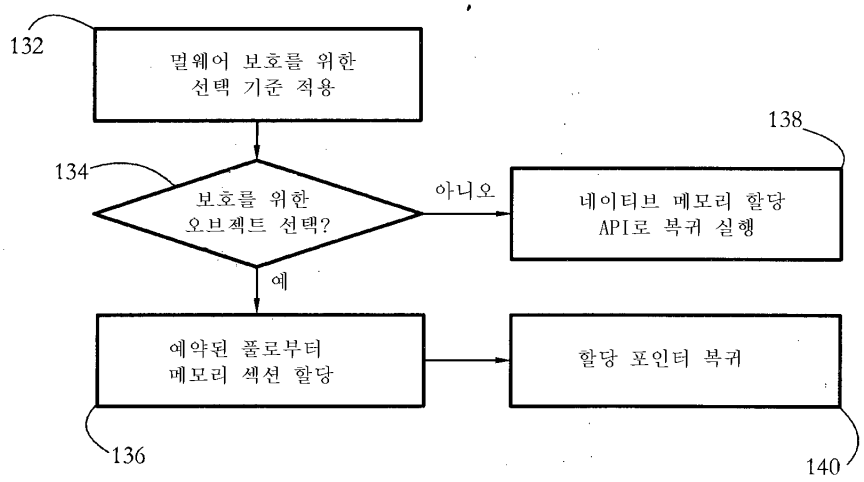
도면7


102
OS 부트-업 검출
104
OS 식별
106
메모리 프라이밍 실행을 위해 보호 프라이밍 모듈 호출
보호를 위한 타겟 오브젝트 선택
108
타겟 오브젝트 초기화?
아니오
110
예
타겟 오브젝트에 할당된 메모리 페이지(들) 보호
112

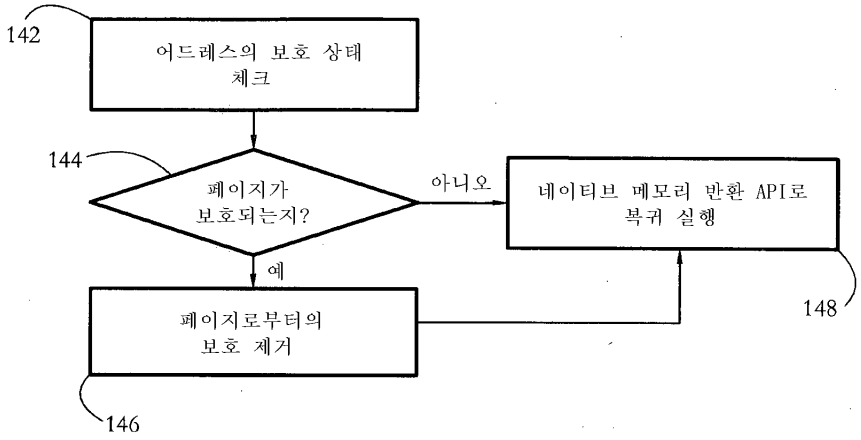
도면8


122
멀웨어 보호를 위한 선택 기준 적용
124
보호를 위한 오브젝트 선택?
아니오
네이티브 메모리 할당 API로 복귀 실행
128
예
오브젝트의 크기를 페이지 크기의 배수로 변경
126

도면9


132
멀웨어 보호를 위한 선택 기준 적용
138
134
보호를 위한 오브젝트 선택?
아니오
네이티브 메모리 할당 API로 복귀 실행
예
예약된 풀로부터 메모리 섹션 할당
할당 포인터 복귀
136
140

도면10


142
어드레스의 보호 상태 체크
144
페이지가 보호되는지?
아니오
네이티브 메모리 반환 API로 복귀 실행
148
예
페이지로부터의 보호 제거
146