(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2025년04월01일
(11) 등록번호 10-2789374
(24) 등록일자 2025년03월27일

(51) 국제특허분류(Int. Cl.)
G06F 12/14 (2006.01) *G06F 12/1009* (2016.01)
G06F 12/1036 (2016.01) *G06F 12/109* (2016.01)
G06F 9/455 (2018.01) *H04L 9/40* (2022.01)
(52) CPC특허분류
G06F 12/1475 (2013.01)
G06F 12/1009 (2013.01)
(21) 출원번호 10-2021-7026646
(22) 출원일자(국제) 2020년03월06일
심사청구일자 2021년08월30일
(85) 번역문제출일자 2021년08월20일
(65) 공개번호 10-2021-0119466
(43) 공개일자 2021년10월05일
(86) 국제출원번호 PCT/EP2020/055979
(87) 국제공개번호 WO 2020/182644
국제공개일자 2020년09월17일
(30) 우선권주장
16/296,450 2019년03월08일 미국(US)
(56) 선행기술조사문헌
US20090307440 A1
(뒷면에 계속)
전체 청구항 수 : 총 25 항

(73) 특허권자
인터내셔널 비지네스 머신즈 코포레이션
미국 10504 뉴욕주 아몬크 뉴오차드 로드
(72) 발명자
헬러, 리사
미국 뉴욕 12601, 포킵시, 사우스 로드 2455, 아이비엠 코포레이션
부사바, 파디
미국 뉴욕 12601, 포킵시, 사우스 로드 2455, 아이비엠 코포레이션
브래드버리, 조나단
미국 뉴욕 12601, 포킵시, 사우스 로드 2455, 아이비엠 코포레이션
(74) 대리인
김태홍, 김진희

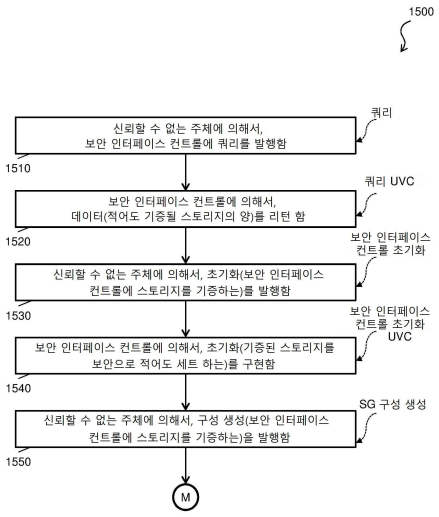
심사관 : 김결

(54) 발명의 명칭 **보안 인터페이스 컨트롤 보안 스토리지 하드웨어 태깅**

(57) 요 약

하나의 방법이 제공된다. 신뢰할 수 없는 주체(untrusted entity)와 통신하는 보안 인터페이스 컨트롤이 상기 방법을 수행한다. 이와 관련하여, 상기 보안 인터페이스 컨트롤은 기부된 스토리지(donated storage)를 보안(secure)으로 세트하기 위한 초기화 명령(an initialization instruction)을 구현한다. 상기 초기화 명령을 구현하는 단계는 상기 신뢰할 수 없는 주체로부터 발행된 명령 호출(an instruction call)에 응답한다.

대 표 도 - 도15



(52) CPC특허분류
G06F 12/1036 (2013.01)
G06F 12/109 (2013.01)
G06F 12/1441 (2013.01)
G06F 12/145 (2013.01)
G06F 9/45558 (2013.01)
H04L 63/10 (2023.05)
G06F 2009/45579 (2019.08)
G06F 2009/45583 (2019.08)
G06F 2009/45587 (2019.08)

(56) 선행기술조사문헌
US20130054528 A1
US20160350018 A1
US20170371803 A1*
US20180285140 A1*
US20160299851 A1*
*는 심사관에 의하여 인용된 문헌

명 세 서

청구범위

청구항 1

방법에 있어서, 상기 방법은:

기부된 스토리지(donated storage)를 시스템의 신뢰할 수 없는 주체(untrusted entity)로부터 보안(secure)으로 세트하기 위해서, 상기 시스템의 보안 인터페이스 컨트롤에 의해서, 초기화 명령(an initialization instruction)을 상기 시스템의 신뢰할 수 없는 주체로부터 수신하는 단계 - 상기 신뢰할 수 없는 주체는 상기 시스템에 하나 또는 그 이상의 가상 머신들을 배치하는 것을 용이하게 하고, 상기 보안 인터페이스 컨트롤은 상기 하나 또는 그 이상의 가상 머신들에 하나 또는 그 이상의 서비스들을 상기 신뢰할 수 없는 주체가 제공하는 것을 용이하게 함 -;

상기 기부된 스토리지의 일부분을 존-특정 스토리지(zone-specific storage)로 할당하는 것(assigning)과 상기 신뢰할 수 없는 주체 및 상기 하나 또는 그 이상의 가상 머신들에 의한 액세스를 금지하는 보안 컨트롤을 위해 상기 존-특정 스토리지를 고유-보안 도메인(a unique-secure domain)으로 태그 하는 것(tagging)을 통해 상기 초기화 명령을, 상기 보안 인터페이스 컨트롤에 의해서, 구현하는 단계; 그리고

상기 보안 인터페이스 컨트롤에 의해서, 보안 게스트-도메인-특정 스토리지(secure guest-domain-specific storage)를 할당하는 단계 및 상기 보안 게스트-도메인-특정 스토리지를 상기 보안 인터페이스 컨트롤에 속함과 관련된 보안-게스트-도메인으로 자격이 있음(qualified with an associated secure-guest-domain)으로 태그하는 단계를 포함하고, 상기 보안-게스트-도메인은 상기 하나 또는 그 이상의 가상 머신들 중 하나에 보안 액세스를 제공하고 상기 신뢰할 수 없는 주체로부터의 액세스는 금지하는

방법.

청구항 2

제1 항에 있어서, 상기 초기화 명령을 구현하는 단계는 상기 보안 인터페이스 컨트롤이 상기 기부된 스토리지를 비-보안으로(as non-secure) 확인하게 하고(verify) 상기 기부된 스토리지를 보안 스토리지로 세트 하게 하는

방법.

청구항 3

제1 항에 있어서, 상기 초기화 명령을 구현하는 단계는 상기 보안 인터페이스 컨트롤이 상기 기부된 스토리지의 일부분을 절대 스토리지로 정의되도록 할당하게 하고 상기 기부된 스토리지의 이 부분을 연관된 가상 주소를 갖지 않는 것으로 태그 하게 하는

방법.

청구항 4

제1 항에 있어서, 상기 초기화 명령을 구현하는 단계는 상기 보안 인터페이스 컨트롤이 상기 기부된 스토리지의 일부분을 가상 스토리지로 정의되도록 할당하게 하고 상기 기부된 스토리지의 이 부분을 연관된 가상 주소로 태그 하게 하는

방법.

청구항 5

제1 항에 있어서,

상기 기부된 스토리지는 상기 보안 인터페이스 컨트롤에 등록되고(registered), 상기 기부된 스토리지는 보안으로 마크되는(marked)

방법.

청구항 6

제1 항에 있어서, 상기 방법은:

상기 기부된 스토리지를 상기 보안 인터페이스 컨트롤에 등록하고(register), 상기 기부된 스토리지를 보안으로 마크하며(mark), 상기 보안 인터페이스 컨트롤이 상기 기부된 스토리지를 상기 보안 인터페이스 컨트롤에 속함으로 태그 하게 하는 구성 생성 명령(a create configuration instruction)을, 상기 보안 인터페이스 컨트롤에 의해서, 구현하는 단계를 더 포함하고, 상기 구성 생성 명령을 구현하는 단계는 상기 신뢰할 수 없는 주체로부터 발행된 제2 명령 호출(a second instruction call)에 응답하는

방법.

청구항 7

제1 항에 있어서, 상기 방법은:

상기 기부된 스토리지를 상기 보안 인터페이스 컨트롤에 등록하고(register), 상기 기부된 스토리지를 보안으로 마크하며(mark), 상기 보안 인터페이스 컨트롤이상기 기부된 스토리지를 상기 보안 인터페이스 컨트롤에 속함과 상기 연관된 보안-게스트-도메인으로 자격이 있음(qualified with the associated secure-guest-domain)으로 태그 하게 하는 중앙 처리 유닛(CPU) 생성 명령(a create central processing unit (CPU) instruction)을, 상기 보안 인터페이스 컨트롤에 의해서, 구현하는 단계를 더 포함하고, 상기 중앙 처리 유닛(CPU) 생성 명령을 구현하는 단계는 상기 신뢰할 수 없는 주체로부터 발행된 제2 명령 호출(a second instruction call)에 응답하는

방법.

청구항 8

제1 항에 있어서, 상기 기부된 스토리지는 상기 신뢰할 수 없는 주체에 의해서 기부된 모든 보안 인터페이스 컨트롤 스토리지를 포함하는

방법.

청구항 9

구현된 프로그램 명령들을 갖는 컴퓨터-판독가능 스토리지 매체에 있어서, 컴퓨터에 의해서 실행 가능한 상기 프로그램 명령들은 연산들을 포함하는 방법을 수행하며, 상기 방법은:

기부된 스토리지(donated storage)를 시스템의 신뢰할 수 없는 주체(untrusted entity)로부터 보안(secure)으로 세트하기 위해서, 상기 시스템의 보안 인터페이스 컨트롤에 의해서, 초기화 명령(an initialization instruction)을 상기 시스템의 신뢰할 수 없는 주체로부터 수신하는 단계 - 상기 신뢰할 수 없는 주체는 상기 시스템에 하나 또는 그 이상의 가상 머신들을 배치하는 것을 용이하게 하고, 상기 보안 인터페이스 컨트롤은 상기 하나 또는 그 이상의 가상 머신들에 하나 또는 그 이상의 서비스들을 상기 신뢰할 수 없는 주체가 제공하는 것을 용이하게 함 -;

상기 기부된 스토리지의 일부분을 존-특정 스토리지(zone-specific storage)로 할당하는 것(assigning)과 상기 신뢰할 수 없는 주체 및 상기 하나 또는 그 이상의 가상 머신들에 의한 액세스를 금지하는 보안 컨트롤을 위해 상기 존-특정 스토리지를 고유-보안 도메인(a unique-secure domain)으로 태그 하는 것(tagging)을 통해 상기 초기화 명령을, 상기 보안 인터페이스 컨트롤에 의해서, 구현하는 단계; 그리고

상기 보안 인터페이스 컨트롤에 의해서, 보안 게스트-도메인-특정 스토리지(secure guest-domain-specific storage)를 할당하는 단계 및 상기 보안 게스트-도메인-특정 스토리지를 상기 보안 인터페이스 컨트롤에 속함과 관련된 보안-게스트-도메인으로 자격이 있음(qualified with an associated secure-guest-domain)으로 태그하는 단계를 포함하고, 상기 보안-게스트-도메인은 상기 하나 또는 그 이상의 가상 머신들 중 하나에 보안 액세스를 제공하고 상기 신뢰할 수 없는 주체로부터의 액세스는 금지하는

컴퓨터-판독가능 스토리지 매체.

청구항 10

제9항에 있어서, 상기 초기화 명령을 구현하는 단계는 상기 보안 인터페이스 컨트롤이 상기 기부된 스토리지를 비-보안으로(as non-secure) 확인하게 하고(verify) 상기 기부된 스토리지를 보안 스토리지로 세트 하게 하는

컴퓨터-판독가능 스토리지 매체.

청구항 11

제9항에 있어서, 상기 초기화 명령을 구현하는 단계는 상기 보안 인터페이스 컨트롤이 상기 기부된 스토리지의 일부분을 절대 스토리지로 정의되도록 할당하게 하고 상기 기부된 스토리지의 이 부분을 연관된 가상 주소를 갖지 않는 것으로 태그 하게 하는

컴퓨터-판독가능 스토리지 매체.

청구항 12

제9항에 있어서, 상기 초기화 명령을 구현하는 단계는 상기 보안 인터페이스 컨트롤이 상기 기부된 스토리지의 일부분을 가상 스토리지로 정의되도록 할당하게 하고 상기 기부된 스토리지의 이 부분을 연관된 가상 주소를 갖는 것으로 태그 하게 하는

컴퓨터-판독가능 스토리지 매체.

청구항 13

제9항에 있어서,

상기 기부된 스토리지는 상기 보안 인터페이스 컨트롤에 등록되고(registered), 상기 기부된 스토리지는 보안으로 마크되는(marked)

컴퓨터-판독가능 스토리지 매체.

청구항 14

제9항에 있어서, 상기 방법은:

상기 기부된 스토리지를 상기 보안 인터페이스 컨트롤에 등록하고(register), 상기 기부된 스토리지를 보안으로 마크하며(mark), 상기 보안 인터페이스 컨트롤이 상기 기부된 스토리지를 상기 보안 인터페이스 컨트롤에 속함으로 태그 하게 하는 구성 생성 명령(a create configuration instruction)을, 상기 보안 인터페이스 컨트롤에 의해서, 구현하는 단계를 더 포함하고, 상기 구성 생성 명령을 구현하는 단계는 상기 신뢰할 수 없는 주체로부터 발행된 제2 명령 호출(a second instruction call)에 응답하는

컴퓨터-판독가능 스토리지 매체.

청구항 15

제9항에 있어서, 상기 방법은:

상기 기부된 스토리지를 상기 보안 인터페이스 컨트롤에 등록하고(register), 상기 기부된 스토리지를 보안으로 마크하며(mark), 상기 보안 인터페이스 컨트롤이 상기 기부된 스토리지를 상기 보안 인터페이스 컨트롤에 속함과 상기 연관된 보안-게스트-도메인으로 자격이 있음으로 태그 하게 하는 중앙 처리 유닛(CPU) 생성 명령(a create central processing unit (CPU) instruction)을, 상기 보안 인터페이스 컨트롤에 의해서, 구현하는 단계를 더 포함하고, 상기 중앙 처리 유닛(CPU) 생성 명령을 구현하는 단계는 상기 신뢰할 수 없는 주체로부터 발행된 제2 명령 호출(a second instruction call)에 응답하는

컴퓨터-판독가능 스토리지 매체.

청구항 16

시스템에 있어서, 상기 시스템은:

메모리;

처리 유닛; 및

상기 메모리 및 상기 처리 유닛과 인터페이스된(interfaced) 보안 인터페이스 컨트롤을 포함하고, 상기 보안 인터페이스 컨트롤은 신뢰할 수 없는 주체(untrusted entity)와 통신하며 연산들(operations)을 수행하도록 구성되며, 상기 연산들은:

기부된 스토리지(donated storage)를 시스템의 신뢰할 수 없는 주체(untrusted entity)로부터 보안(secure)으로 세트하기 위해서, 상기 시스템의 보안 인터페이스 컨트롤에 의해서, 초기화 명령(an initialization instruction)을 상기 시스템의 신뢰할 수 없는 주체로부터 수신하는 단계 - 상기 신뢰할 수 없는 주체는 상기 시스템에 하나 또는 그 이상의 가상 머신들을 배치하는 것을 용이하게 하고, 상기 보안 인터페이스 컨트롤은 상기 하나 또는 그 이상의 가상 머신들에 하나 또는 그 이상의 서비스들을 상기 신뢰할 수 없는 주체가 제공하는 것을 용이하게 함 -;

상기 기부된 스토리지의 일부분을 존-특정 스토리지(zone-specific storage)로 할당하는 것(assigning)과 상기 신뢰할 수 없는 주체 및 상기 하나 또는 그 이상의 가상 머신들에 의한 액세스를 금지하는 보안 컨트롤을 위해 상기 존-특정 스토리지를 고유-보안 도메인(a unique-secure domain)으로 태그 하는 것(tagging)을 통해 상기 초기화 명령을, 상기 보안 인터페이스 컨트롤에 의해서, 구현하는 단계; 그리고

상기 보안 인터페이스 컨트롤에 의해서, 보안 게스트-도메인-특정 스토리지(secure guest-domain-specific storage)를 할당하는 단계 및 상기 보안 게스트-도메인-특정 스토리지를 상기 보안 인터페이스 컨트롤에 속함과 관련된 보안-게스트-도메인으로 자격이 있음(qualified with an associated secure-guest-domain)으로 태그하는 단계를 포함하고, 상기 보안-게스트-도메인은 상기 하나 또는 그 이상의 가상 머신들 중 하나에 보안 액세스를 제공하고 상기 신뢰할 수 없는 주체로부터의 액세스는 금지하는

시스템.

청구항 17

제16항에 있어서, 상기 초기화 명령을 구현하는 단계는 상기 보안 인터페이스 컨트롤이 상기 기부된 스토리지를 비-보안으로(as non-secure) 확인하게 하고(verify) 상기 기부된 스토리지를 보안 스토리지로 세트 하게 하는

시스템.

청구항 18

제16항에 있어서, 상기 초기화 명령을 구현하는 단계는 상기 보안 인터페이스 컨트롤이 상기 기부된 스토리지의 일부분을 절대 스토리지로 정의되도록 할당하게 하고 상기 기부된 스토리지의 이 부분을 연관된 가상 주소를 갖지 않는 것으로 태그 하게 하는

시스템.

청구항 19

제16항에 있어서, 상기 보안 인터페이스 컨트롤은 연산들을 수행하도록 구성되고, 상기 연산들은:

상기 기부된 스토리지를 상기 보안 인터페이스 컨트롤에 등록하고(register), 상기 기부된 스토리지를 보안으로 마크하며(mark), 상기 보안 인터페이스 컨트롤이 상기 기부된 스토리지를 상기 보안 인터페이스 컨트롤에 속함으로 태그 하게 하는 구성 생성 명령(a create configuration instruction)을 구현하는 단계를 포함하고, 상기 구성 생성 명령을 구현하는 단계는 상기 신뢰할 수 없는 주체로부터 발행된 제2 명령 호출(a second instruction call)에 응답하는

시스템.

청구항 20

제16항에 있어서, 상기 보안 인터페이스 컨트롤은 연산들을 수행하도록 구성되고, 상기 연산들은:

상기 기부된 스토리지를 상기 보안 인터페이스 컨트롤에 등록하고(register), 상기 기부된 스토리지를 보안으로 마크하며(mark), 상기 보안 인터페이스 컨트롤이 상기 기부된 스토리지를 상기 보안 인터페이스 컨트롤에 속함과 상기 연관된 보안-게스트-도메인으로 자격이 있음으로 태그 하게 하는 중앙 처리 유닛(CPU) 생성 명령(a

create central processing unit (CPU) instruction)을 구현하는 단계를 포함하고, 상기 중앙 처리 유닛(CPU) 생성 명령을 구현하는 단계는 상기 신뢰할 수 없는 주체로부터 발행된 제2 명령 호출(a second instruction call)에 응답하는

시스템.

청구항 21

방법에 있어서, 상기 방법은:

기부된 스토리지를 보안 인터페이스 컨트롤에 등록하고(register), 상기 기부된 스토리지를 보안으로 마크하는(mark) 보안 인터페이스 컨트롤 초기화 명령(an initialize secure interface control instruction)을, 시스템의 신뢰할 수 없는 주체(untrusted entity in a system)와 통신하는 상기 보안 인터페이스 컨트롤에 의해서, 구현하는 단계 - 상기 보안 인터페이스 컨트롤 초기화 명령을 구현하는 단계는 상기 신뢰할 수 없는 주체로부터 발행된 제1 명령 호출(a first instruction call)에 응답하고, 상기 신뢰할 수 없는 주체는 상기 시스템에 하나 또는 그 이상의 가상 머신들을 배치하는 것을 용이하게 하고, 상기 보안 인터페이스 컨트롤은 상기 하나 또는 그 이상의 가상 머신들에 하나 또는 그 이상의 서비스들을 상기 신뢰할 수 없는 주체가 제공하는 것을 용이하게 함 -;

상기 기부된 스토리지를 상기 보안 인터페이스 컨트롤에 등록하고(register), 상기 기부된 스토리지를 보안으로 마크하는(mark) 보안 구성 생성 명령(a create secure configuration instruction)을, 상기 보안 인터페이스 컨트롤에 의해서, 구현하는 단계 - 상기 보안 구성 생성 명령을 구현하는 단계는 상기 하나 또는 그 이상의 가상 머신들 중 하나에 보안 액세스를 제공하고 상기 신뢰할 수 없는 주체로부터의 액세스는 금지하도록 상기 신뢰할 수 없는 주체로부터 발행된 제2 명령 호출(a second instruction call)에 응답함 -; 및

상기 기부된 스토리지를 상기 보안 인터페이스 컨트롤에 등록하고(register), 상기 기부된 스토리지를 보안으로 마크하는(mark) 보안 중앙 처리 유닛(CPU) 생성 명령(a create secure central processing unit (CPU) instruction)을, 상기 보안 인터페이스 컨트롤에 의해서, 구현하는 단계를 포함하고, 상기 중앙 처리 유닛(CPU) 생성 명령을 구현하는 단계는 상기 하나 또는 그 이상의 가상 머신들 중 하나를 위해 하나 또는 그 이상의 가상 CPU들을 생성하고 상기 신뢰할 수 없는 주체에 의한 액세스는 금지하도록 상기 신뢰할 수 없는 주체로부터 발행된 제3 명령 호출(a third instruction call)에 응답하는

방법.

청구항 22

제21항에 있어서,

상기 보안 인터페이스 컨트롤 초기화 명령을 구현하는 단계는 상기 보안 인터페이스 컨트롤이 상기 기부된 스토리지를 존-특정 스토리지(zone-specific storage)로 할당하게 하고 상기 존-특정 스토리지를 고유-보안 도메인(a unique-secure domain)으로 태그 하게 하며,

상기 보안 구성 생성 명령을 구현하는 단계는 상기 보안 인터페이스 컨트롤이 기부된 베이스-보안-구성 및 가변-보안-구성 스토리지(donated base-secure-configuration and variable-secure-configuration storage)를 보안 게스트-도메인-특정 스토리지(secure guest-domain-specific storage)로서 할당하게 하고 상기 보안 게스트-도메인-특정 스토리지를 상기 보안 인터페이스 컨트롤에 속함과 연관된 보안-게스트-도메인 자격이 있음으로 태그 하게 하며, 또는

상기 중앙 처리 유닛(CPU) 생성 명령을 구현하는 단계는 상기 보안 인터페이스 컨트롤이 기부된 베이스-보안-CPU 스토리지(donated base-secure-CPU storage)를 보안 게스트-도메인-특정 스토리지(secure guest-domain-specific storage)로 할당하게 하고 상기 보안 게스트-도메인-특정 스토리지를 상기 보안 인터페이스 컨트롤에 속함과 상기 관련된 보안-게스트-도메인 자격 있음으로 태그 하게 하는

방법.

청구항 23

제22항에 있어서, 상기 기부된 존-특정 스토리지, 상기 베이스-보안-구성 스토리지, 또는 상기 베이스-보안-CPU 스토리지는 절대 스토리지(absolute storage)로서 정의되고 연관된 가상주소를 갖지 않음으로 태그 되며, 그리

고

상기 기부된 가변-보안-구성 스토리지는 가상 스토리지로서 정의되고 상기 연관된 가상 주소로 태그 되는

방법.

청구항 24

제21항에 있어서, 상기 보안 인터페이스 컨트롤 초기화 명령, 상기 보안 구성 생성 명령, 또는 상기 보안 중앙 처리 유닛(CPU) 생성 명령은 상기 보안 인터페이스 컨트롤이 기부된 스토리지가 비-보안으로 확인하게 하고 상기 기부된 스토리지를 보안으로 세트 하게 하는

방법.

청구항 25

제21항에 있어서, 상기 기부된 스토리지는 상기 신뢰할 수 없는 주체에 의해서 기부된 모든 보안 인터페이스 컨트롤 스토리지를 포함하는

방법.

발명의 설명

기 술 분 야

[0001] [0001] 본 발명은 일반적으로 컴퓨터 기술에 관한 것이고, 더욱 구체적으로 보안 인터페이스 컨트롤 보안 스토리지 하드웨어 태깅(secure interface control secure storage hardware tagging)에 관한 것이다.

배 경 기 술

[0002] [0002] 클라우드 컴퓨팅 및 클라우드 스토리지는 사용자에게 제3자의 데이터 센터들에 데이터를 저장하고 처리할 수 있는 능력을 제공한다. 클라우드 컴퓨팅은 고객에게 하드웨어를 구입하거나 물리적 서버를 위한 바닥 공간을 제공할 것을 요구하지 않으면서 가상 머신(VM)을 빠르고 쉽게 공급하는 능력을 제공한다. 고객은 고객의 선호도 혹은 요구 사항의 변화에 따라 VM을 쉽게 확장하거나 축소할 수 있다. 일반적으로 클라우드 컴퓨팅 공급자는 공급자의 데이터 센터에 있는 서버에 물리적으로 상주하는 VM을 공급한다(provision). 특히 컴퓨팅 공급자가 동일한 서버에 둘 이상의 고객 데이터를 저장하는 경우가 많기 때문에 고객은 종종 VM에서의 데이터 보안(the security of data)에 대해 걱정한다. 고객은 자신의 코드/데이터와 클라우드 컴퓨팅 공급자의 코드/데이터 사이, 그리고 자신의 코드/데이터와 공급자의 사이트에서 실행되는 다른 VM의 코드/데이터 간의 보안을 원할 수 있다. 또한 고객은 공급자의 컴퓨터에서 실행되는 다른 코드로부터의 잠재적인 보안 침해(potential security breaches from other code running on the machine)뿐만 아니라 공급자의 관리자로부터 보안을 원할 수 있다.

발명의 내용

해결하려는 과제

[0003] [0003] 그러한 민감한 상황을 처리하기 위해 클라우드 서비스 공급자는 적절한 데이터 격리(data isolation)및 논리 스토리지 분리(logical storage segregation)를 보장하기 위해 보안 컨트롤들을 구현할 수 있다. 클라우드 인프라를 구현할 때 가상화를 광범위하게 사용하면 가상화가 운영 체제(OS)와 기본(underlying) 하드웨어(그 것은 컴퓨팅, 스토리지 또는 네트워킹 하드웨어일 수 있다)간의 관계를 변경하므로 클라우드 서비스 고객에게 고유한 보안 문제가 발생한다. 이 때문에 자체적으로 적절하게 구성, 관리 및 보안이 되어야 하는 추가 계층으로서 가상화가 도입된다.

[0004] [0004] 일반적으로, 호스트 하이퍼바이저의 컨트롤 하에서 하나의 게스트로서 실행되는, VM은, 상기 게스트에 대한 가상화 서비스들을 투명하게 제공하기 위해 상기 하이퍼바이저에 의존한다. 이들 서비스들은 메모리 관리, 명령 에뮬레이션 및 인터럽트 처리를 포함한다.

[0005] [0005] 메모리 관리의 경우, VM은 디스크로부터 데이터를 이동시켜(페이지-인(page-in)) 메모리에 상주하게 할 수 있고, VM은 또한 디스크로 데이터를 다시 이동시킬(페이지-아웃(page-out)) 수도 있다. 페이지가 메모리에 상주하는 동안, VM(게스트)은 동적 주소 변환(dynamic address translation: DAT)을 사용하여 메모리의 페이지

들을 게스트 가상 주소(a guest virtual address)로부터 게스트 절대 주소(a guest absolute address)로 맵핑한다(map). 또한, 호스트 하이퍼바이저는 메모리의 게스트 페이지에 대한 자체 DAT 매핑(호스트 가상 주소로부터 호스트 절대 주소로)을 갖고 있으며, 그리고 하이퍼바이저는 게스트에 대해 독립적이고 투명하게 게스트 페이지들을 메모리에 페이지-인 하고 메모리로부터 페이지-아웃할 수 있다. 하이퍼바이저가 두 개의 개별 게스트 VM들 간에 게스트 메모리 공유 또는 메모리 격리(memory isolation or sharing of guest memory)를 제공하는 것은 호스트 DAT 테이블을 통해서이다. 또한 호스트는 게스트 메모리에 액세스하여, 필요한 경우, 게스트를 대신하여 게스트 연산들을 시뮬레이션 할 수 있다(simulate).

과제의 해결 수단

[0006] [0006] 본 발명의 하나 또는 그 이상의 실시 예들에 따라, 방법이 제공된다. 상기 방법은 기부된 스토리지(donated storage)를 보안(secure)으로 세트하기 위해서, 초기화 명령(an initialization instruction)을, 신뢰할 수 없는 주체(untrusted entity)와 통신하는 보안 인터페이스 컨트롤에 의해서, 구현하는 단계를 포함하고, 상기 초기화 명령을 구현하는 단계는 상기 신뢰할 수 없는 주체로부터 발행된 명령 호출(an instruction call)에 응답한다. 본 발명의 하나 또는 이상의 실시 예들의 기술적 효과들과 잇점들은 상기 보안 인터페이스 컨트롤 내에 추가의 보안을 허용하는, 이러한 태깅 기술을 포함한다는 것이다. 이론상(by definition) 보안 인터페이스 컨트롤은 모든 스토리지에 액세스할 수 있지만 실제로는(by design) 보안 인터페이스 컨트롤은 이 스토리지의 각 부분(보안 게스트 스토리지 뿐 만 아니라 보안 인터페이스 컨트롤의 별도 부분들)에 신중하게 액세스하는데, 이는 하드웨어가 이들 액세스들에 관하여 보안 체크들을 시행할 수 있게 하기 위함이다.

[0007] [0007] 본 발명의 하나 또는 그 이상의 실시 예들 또는 위의 방법 실시예에 따라, 상기 초기화 명령을 구현하는 단계는 상기 보안 인터페이스 컨트롤이 상기 기부된 스토리지의 일부분을 존-특정 스토리지(zone-specific storage)로 할당하게 하고 상기 존-특정 스토리지를 고유-보안 도메인(a unique-secure domain)으로 태그(tag)하게 할 수 있다.

[0008] [0008] 본 발명의 하나 또는 그 이상의 실시 예들 또는 위의 방법 실시예에 따라, 상기 초기화 명령을 구현하는 단계는 상기 보안 인터페이스 컨트롤이 상기 기부된 스토리지의 일부분을 보안 게스트-도메인-특정 스토리지(secure guest-domain-specific storage)로 할당하게 하고 상기 보안 게스트-도메인-특정 스토리지가 상기 보안 인터페이스 컨트롤에 속함과 관련된 보안-게스트-도메인으로 자격이 있음(qualified with the associated secure-guest-domain)으로 태그 하게 할 수 있다.

[0009] [0009] 본 발명의 하나 또는 그 이상의 실시 예들 또는 위의 방법 실시예들 중 어느 하나에 따라, 상기 초기화 명령을 구현하는 단계는 상기 보안 인터페이스 컨트롤이 상기 기부된 스토리지를 비-보안으로(as non-secure) 확인하게 하고(verify) 상기 기부된 스토리지를 보안 스토리지로 세트 하게 할 수 있다.

[0010] [0010] 본 발명의 하나 또는 그 이상의 실시 예들 또는 위의 방법 실시예들 중 어느 하나에 따라, 상기 초기화 명령을 구현하는 단계는 상기 보안 인터페이스 컨트롤이 상기 기부된 스토리지의 일부분을 절대 스토리지로 정의되도록 할당하게 하고 상기 기부된 스토리지의 이 부분을 연관된 가상 주소를 갖지 않는 것으로 태그 하게 할 수 있다.

[0011] [0011] 본 발명의 하나 또는 그 이상의 실시 예들 또는 위의 방법 실시예들 중 어느 하나에 따라, 상기 초기화 명령을 구현하는 단계는 상기 보안 인터페이스 컨트롤이 상기 기부된 스토리지의 일부분을 가상 스토리지로 정의되도록 할당하게 하고 상기 기부된 스토리지의 이 부분을 연관된 가상 주소로 태그 하게 할 수 있다.

[0012] [0012] 본 발명의 하나 또는 그 이상의 실시 예들 또는 위의 방법 실시예들 중 어느 하나에 따라, 상기 방법은 상기 기부된 스토리지를 상기 보안 인터페이스 컨트롤에 등록하고(register), 상기 기부된 스토리지를 보안으로 마크하며(mark), 상기 보안 인터페이스 컨트롤이 상기 기부된 스토리지를 상기 기부된 스토리지를 상기 보안 인터페이스 컨트롤에 속함과 연관된 보안-게스트-도메인으로 자격이 있음으로 태그 하게 하는 보안 인터페이스 컨트롤 초기화 명령을, 상기 보안 인터페이스 컨트롤에 의해서, 구현하는 단계를 더 포함할 수 있고, 상기 구성 생성 명령을 구현하는 단계는 상기 신뢰할 수 없는 주체로부터 발행된 제2 명령 호출에 응답할 수 있다. 보안 페이지를 마크하는 것의 기술적 효과들과 잇점들은 모든 비-보안 주체들에 의한 액세스를 금지시키는 것을 포함한다는 것이다.

[0013] [0013] 본 발명의 하나 또는 그 이상의 실시 예들 또는 위의 방법 실시예들 중 어느 하나에 따라, 상기 방법은 상기 기부된 스토리지를 상기 보안 인터페이스 컨트롤에 등록하고(register), 상기 기부된 스토리지를 보안으로

마크하며(mark), 상기 보안 인터페이스 컨트롤이 상기 기부된 스토리지를 상기 보안 인터페이스 컨트롤에 속함으로 태그 하게 하는 구성 생성 명령(a create configuration instruction)을, 상기 보안 인터페이스 컨트롤에 의해서, 구현하는 단계를 더 포함할 수 있고, 상기 구성 생성 명령을 구현하는 단계는 상기 신뢰할 수 없는 주체로부터 발행된 제2 명령 호출(a second instruction call)에 응답할 수 있다. 보안 페이지를 마크하는 것의 기술적 효과들과 잇점들은 모든 비-보안 주체들에 의한 액세스를 금지시키는 것을 포함한다는 것이다.

[0014] [0014] 본 발명의 하나 또는 그 이상의 실시 예들 또는 위의 방법 실시예들 중 어느 하나에 따라, 상기 방법은 상기 기부된 스토리지를 상기 보안 인터페이스 컨트롤에 등록하고(register), 상기 기부된 스토리지를 보안으로 마크하며(mark), 상기 보안 인터페이스 컨트롤이상기 기부된 스토리지를 상기 보안 인터페이스 컨트롤에 속함과 연관된 보안-게스트-도메인으로 자격이 있음(qualified with associated secure-guest-domain)으로 태그 하게 하는 중앙 처리 유닛(CPU) 생성 명령(a create central processing unit (CPU) instruction)을, 상기 보안 인터페이스 컨트롤에 의해서, 구현하는 단계를 더 포함할 수 있고, 상기 구성 생성 명령을 구현하는 단계는 상기 신뢰할 수 없는 주체로부터 발행된 제2 명령 호출(a second instruction call)에 응답할 수 있다. 보안 페이지를 마크하는 것의 기술적 효과들과 잇점들은 모든 비-보안 주체들에 의한 액세스를 금지시키는 것을 포함한다는 것이다.

[0015] [0015] 본 발명의 하나 또는 그 이상의 실시 예들 또는 위의 방법 실시예들 중 어느 하나에 따라, 상기 기부된 스토리지는 상기 신뢰할 수 없는 주체에 의해서 기부된 모든 보안 인터페이스 컨트롤 스토리지를 포함할 수 있다.

[0016] [0016] 본 발명의 하나 또는 그 이상의 실시 예들에 따라, 위의 방법 실시예들 중 어느 것도 컴퓨터 프로그램 제품 또는 시스템으로 구현될 수 있다.

[0017] [0017] 본 발명의 하나 또는 그 이상의 실시 예들에 따라, 방법은 기부된 스토리지를 보안 인터페이스 컨트롤에 등록하고(register), 상기 기부된 스토리지를 보안으로 마크하는(mark) 보안 인터페이스 컨트롤 초기화 명령(an initialize secure interface control instruction)을, 신뢰할 수 없는 주체(untrusted entity)와 통신하는 상기 보안 인터페이스 컨트롤에 의해서, 구현하는 단계를 포함하고, 상기 보안 인터페이스 컨트롤 초기화 명령을 구현하는 단계는 상기 신뢰할 수 없는 주체로부터 발행된 제1 명령 호출(a first instruction call)에 응답한다. 상기 방법은 또한 상기 기부된 스토리지를 상기 보안 인터페이스 컨트롤에 등록하고(register), 상기 기부된 스토리지를 보안으로 마크하는(mark) 보안 구성 생성 명령(a create secure configuration instruction)을, 상기 보안 인터페이스 컨트롤에 의해서, 구현하는 단계 - 상기 보안 구성 생성 명령을 구현하는 단계는 상기 신뢰할 수 없는 주체로부터 발행된 제2 명령 호출(a second instruction call)에 응답함 -; 및 상기 기부된 스토리지를 상기 보안 인터페이스 컨트롤에 등록하고(register), 상기 기부된 스토리지를 보안으로 마크하는(mark) 중앙 처리 유닛(CPU) 생성 명령(a create central processing unit (CPU) instruction)을, 상기 보안 인터페이스 컨트롤에 의해서, 구현하는 단계를 포함하고, 상기 중앙 처리 유닛(CPU) 생성 명령을 구현하는 단계는 상기 신뢰할 수 없는 주체로부터 발행된 제3 명령 호출(a third instruction call)에 응답한다. 상기 방법의 기술적 효과들과 잇점들은 상기 보안 인터페이스 컨트롤 내에 추가의 보안을 허용하는, 이러한 태깅 기술을 포함한다는 것이다. 이론상(by definition) 보안 인터페이스 컨트롤은 모든 스토리지에 액세스할 수 있지만 실제로는(by design) 보안 인터페이스 컨트롤은 이 스토리지의 각 부분(보안 게스트 스토리지 뿐 만 아니라 보안 인터페이스 컨트롤의 별도 부분들)에 신중하게 액세스하는데, 이는 하드웨어가 이들 액세스들에 관하여 보안 체크들을 시행할 수 있게 하기 위함이다.

[0018] [0018] 본 발명의 하나 또는 그 이상의 실시 예들 또는 위의 방법 실시예에 따라, 상기 보안 인터페이스 컨트롤 초기화 명령을 구현하는 단계는 상기 보안 인터페이스 컨트롤이 상기 기부된 스토리지를 존-특정 스토리지(zone-specific storage)로 할당하게 할 수 있고 상기 존-특정 스토리지를 고유-보안 도메인(a unique-secure domain)으로 태그 하게 할 수 있으며, 상기 보안 구성 생성 명령을 구현하는 단계는 상기 보안 인터페이스 컨트롤이 기부된 베이스-보안-구성 및 가변-보안-구성 스토리지(donated base-secure-configuration and variable-secure-configuration storage)를 보안 게스트-도메인-특정 스토리지(secure guest-domain-specific storage)로서 할당하게 할 수 있고 상기 보안 게스트-도메인-특정 스토리지를 상기 보안 인터페이스 컨트롤에 속함과 상기 연관된 보안-게스트-도메인 자격이 있음으로 태그 하게 할 수 있으며, 또는 상기 중앙 처리 유닛(CPU) 생성 명령을 구현하는 단계는 상기 보안 인터페이스 컨트롤이 기부된 베이스-보안-CPU 스토리지(donated base-secure-CPU storage)를 보안 게스트-도메인-특정 스토리지(secure guest-domain-specific storage)로 할당하게 할 수 있고 상기 보안 게스트-도메인-특정 스토리지를 상기 보안 인터페이스 컨트롤에 속함과 상기 관련된 보안

-게스트-도메인 자격 있음으로 태그 하게 할 수 있다.

[0019] [0019] 본 발명의 하나 또는 그 이상의 실시 예들 또는 위의 방법 실시예들 중 어느 하나에 따라, 상기 기부된 존-특정 스토리지, 상기 베이스-보안-구성 스토리지, 또는 상기 베이스-CPU 스토리지는 절대 스토리지(absolute storage)로서 정의될 수 있고 연관된 가상주소를 갖지 않음으로 태그 될 수 있으며, 그리고 상기 기부된 가변-보안-구성 스토리지는 가상 스토리지로서 정의되고 연관된 가상 주소로 태그 될 수 있다.

[0020] [0020] 본 발명의 하나 또는 그 이상의 실시 예들 또는 위의 방법 실시예들 중 어느 하나에 따라, 상기 보안 인터페이스 컨트롤 초기화 명령, 상기 보안 구성 생성 명령, 또는 상기 보안 CPU 생성 명령은 상기 보안 인터페이스 컨트롤이 기부된 스토리지가 비-보안으로 확인하게 할 수 있고 상기 기부된 스토리지를 보안으로 세트 하게 할 수 있다.

[0021] [0021] 본 발명의 하나 또는 그 이상의 실시 예들 또는 위의 방법 실시예들 중 어느 하나에 따라, 상기 기부된 스토리지는 상기 신뢰할 수 없는 주체에 의해서 기부된 모든 보안 인터페이스 컨트롤 스토리지를 포함할 수 있다.

[0022] [0022] 본 발명의 하나 또는 그 이상의 실시 예들에 따라, 위의 방법 실시예들 중 어느 것도 컴퓨터 프로그램 제품 또는 시스템으로 구현될 수 있다.

[0023] [0023] 추가적인 특징 및 장점들은 본 발명의 기술을 통해 실현된다. 본 발명의 다른 실시예들 및 특징들은 본 명세서에 상세히 설명되어 있으며 본 발명의 일부로 간주된다. 장점과 특징이 있는 본 발명을 더 잘 이해하려면 상세한 설명과 도면을 참조한다.

도면의 간단한 설명

[0024] [0024] 본 발명의 배타적 권리에 관한 구체적인 사항들은 본원 명세서의 결론 부분에 기재된 청구항들에 구체적으로 지적되고 명확하게 청구된다. 본 발명의 실시예들의 전술한 및 기타 특징들 및 잇점들은 첨부 도면과 함께 제공된 다음의 상세한 설명으로부터 명백하다:

[0025] [0025] 도 1은 본 발명의 하나 또는 그 이상의 실시예들에 따른 존 보안(zone security)을 위한 테이블을 도시한다.

[0026] [0026] 도 2는 본 발명의 하나 또는 그 이상의 실시예들에 따라 DAT를 수행하기 위한 가상 및 절대 주소 공간을 도시한다.

[0027] [0027] 도 3은 본 발명의 하나 또는 그 이상의 실시예들에 따른 하이퍼바이저 하에서 실행되는 가상 머신(VM)을 지원하기 위한 네스트된 멀티-파트 DAT를 도시한다.

[0028] [0028] 도 4는 본 발명의 하나 또는 그 이상의 실시예들에 따른 보안 게스트 스토리지의 매핑을 도시한다.

[0029] [0029] 도 5는 본 발명의 하나 이상의 실시 예들에 따른 동적 주소 변환(DAT) 연산의 시스템 개략도를 도시한다.

[0030] [0030] 도 6은 본 발명의 하나 이상의 실시 예들에 따른 보안 인터페이스 컨트롤 메모리의 시스템 개략도를 도시한다.

[0031] [0031] 도 7는 본 발명의 하나 또는 그 이상의 실시예들에 따른 임포트 연산의 프로세스 플로를 도시한다.

[0032] [0032] 도 8은 본 발명의 하나 또는 그 이상의 실시예들에 따른 임포트 연산의 프로세스 플로를 도시한다.

[0033] [0033] 도 9는 본 발명의 하나 이상의 실시예에 따른 기부된 메모리 연산의 프로세스를 도시한다.

[0034] [0034] 도 10은 본 발명의 하나 이상의 실시예에 따른 보안 인터페이스 컨트롤의 보안 페이지로의 비-보안 하이퍼바이저 페이지의 전환 프로세스 플로를 도시한다.

[0035] [0035] 도 11은 본 발명의 하나 이상의 실시예에 따른 보안 인터페이스 컨트롤에 의해 이루어진 보안 스토리지 액세스의 프로세스 플로를 도시한다.

[0036] [0036] 도 12는 본 발명의 하나 또는 그 이상의 실시예들에 따른 보안 인터페이스 컨트롤에 의한 액세스 태깅의 프로세스 플로를 도시한다.

[0037] [0037] 도 13은 본 발명의 하나 또는 그 이상의 실시예들에 따른 보안 인터페이스 컨트롤에 의한 및 프로그램에

의한 보안 및 비-보안 액세스를 지원하기 위한 변환의 프로세스 플로를 도시한다.

[0038] 도 14는 본 발명의 하나 이상의 실시 예들에 따른 프로그램 및 보안 인터페이스 컨트롤에 의한 보안 스토리지 보호를 갖는 DAT의 프로세스 플로를 도시한다.

[0039] 도 15는 본 발명의 하나 이상의 실시 예들에 따른 보안 인터페이스 컨트롤 보안 스토리지 하드웨어 태깅을 위한 프로세스 플로를 도시한다.

[0040] 도 16은 본 발명의 하나 이상의 실시 예들에 따른, 도 15의 프로세스 플로의 계속인, 보안 인터페이스 컨트롤 보안 스토리지 하드웨어 태깅을 위한 프로세스 플로를 도시한다.

[0041] 도 17은 본 발명의 하나 또는 그 이상의 실시예들에 따른 클라우드 컴퓨팅 환경을 도시한다.

[0042] 도 18는 본 발명의 하나 또는 그 이상의 실시예들에 따른 추상화 모델 계층을 도시한다.

[0043] 도 19은 본 발명의 하나 또는 그 이상의 실시예들에 따른 시스템을 도시한다. 그리고

[0044] 도 20는 본 발명의 하나 또는 그 이상의 실시예들에 따른 노드를 도시한다.

[0045] 본 발명에 도시된 다이어그램은 예시적이다. 본 발명의 정신을 벗어남이 없이 도면 또는 거기에 설명된 동작들에 대한 많은 변형들이 있을 수 있다. 예를 들어 작업들을 다른 순서로 수행하거나 작업을 추가, 삭제 또는 수정할 수 있다. 또한, "결합된"이라는 용어 및 그 변형들은 2개의 엘리멘트들 사이에 통신 경로를 갖는 것을 설명하며, 엘리멘트들 사이에 개재 엘리멘트/연결이 없는 엘리멘트들 사이의 직접적인 연결을 의미하지 않는다. 이러한 모든 변형들은 본 명세서의 일부로 간주된다.

발명을 실시하기 위한 구체적인 내용

[0025] [0046] 본 발명의 하나 또는 그 이상의 실시 예들은 보안 인터페이스 컨트롤 메모리를 보안으로 표시하고 보안 인터페이스 컨트롤 이외의 모든 주체에 의해서 상기 보안 인터페이스 컨트롤 메모리가 액세스될 수 없도록 그것을 태그 하는 (tag) 태깅 메커니즘을 제공한다. 보안 주체들 간의 격리를 위해 하드웨어에 제공된 동일 스토리지 보안 메커니즘들이 보안 인터페이스 컨트롤 스토리지의 격리를 위해 또한 사용된다.

[0026] [0047] 호스트 하이퍼바이저 컨트롤 하의 게스트로서 실행되는, 가상 머신(VM)은 상기 게스트에 대한 가상화 서비스를 투명하게 제공하기 위해 상기 하이퍼바이저에 의존한다. 이들 서비스들은 보안 주체와 보안 자원들에 대하여 이 다른 주체에 의한 액세스를 전통적으로 허용하는 다른 신뢰할 수 없는 주체 사이의 모든 인터페이스에 대해서 적용할 수 있다. 앞서 언급했듯이, 이들 서비스들에는 메모리 관리, 명령 에뮬레이션, 그리고 인터럽트 처리가 포함될 수 있지만 이에 국한되지는 않는다. 예를 들어, 인터럽트와 예외 주입(interrupt and exception injection)의 경우, 하이퍼바이저는 일반적으로 게스트의 프리픽스 영역(a prefix area)(로우 코어)을 읽거나 쓴다(reads and/or writes into). 본 명세서에서 사용된 "가상 머신" 또는 "VM"이라는 용어는 물리적 머신(컴퓨팅 디바이스, 프로세서 등) 및 처리 환경(운영 체제 OS, 소프트웨어 자원들 등)의 논리적 표현(a logical representation)을 뜻한다. VM은 기본 호스트 머신(an underlying host machine)(물리적 프로세서 또는 프로세서들의 집합)에서 실행되는 소프트웨어로서 유지 관리된다. 사용자 또는 소프트웨어 자원의 관점에서, VM은 자체적으로 독립적인 물리적 머신인 것처럼 보인다. 본 명세서에서 사용된 "하이퍼바이저" 및 "VM 모니터(VMM)"라는 용어는 동일한 호스트 시스템에서 여러(때로는 다른) OS들을 사용하여 여러 VM들을 관리하고 실행할 수 있도록 하는 처리 환경 또는 플랫폼 서비스를 뜻한다. VM을 배포하는 것은 VM의 설치 프로세스와 VM의 활성화(또는 시작) 프로세스를 포함한다는 것을 이해해야 한다. 다른 실시 예에서, VM을 배포하는 것은 VM의 활성화(또는 시작) 프로세스를 포함한다 (예: VM이 이전에 설치되었거나 이미 존재하는 경우).

[0027] [0048] 보안 게스트들을 용이하게 하고 지원하기 위해서는, 기술적인 문제가 존재하는데, 이는, 하이퍼바이저가 VM으로부터 데이터를 액세스할 수 없고, 따라서, 위에서 설명된 방법으로 서비스들을 제공할 수 없도록 하는, 추가의 보안이 하이퍼바이저에 의존함이 없이 하이퍼바이저와 보안 게스트들 사이에 요구되기 때문이다.

[0028] [0049] 여기서 설명된 보안 실행은 보안 스토리지와 비-보안 스토리지 사이에서뿐만 아니라 다른 보안 사용자들에 속하는 보안 스토리지 사이에서의 격리(isolation)를 보장하는 하드웨어 메커니즘을 제공한다. 보안 게스트들을 위해, "신뢰할 수 없는" 비-보안 하이퍼바이저와 보안 게스트들 간에 추가 보안이 제공된다. 이렇게 하려면, 하이퍼바이저가 일반적으로 게스트들을 대신하여 수행하는 많은 기능들을 머신에 통합해야 한다. 본 명세서에서 "UV"라고도 하는, 새로운 보안 인터페이스 컨트롤은 하이퍼바이저와 보안 게스트들 사이의 보안 인터페이스를 제공하기 위해 설명된다. 보안 인터페이스 컨트롤 및 UV라는 용어는 본 명세서에서 상호 교환적으로 사

용된다. 보안 인터페이스 컨트롤은 하드웨어와 협력하여 이러한 추가 보안을 제공한다.

[0029] [0050] 이 보호 메커니즘은 가상 머신 디스패치들을 전환의 주요 지점(the main point of transition)으로 사용하는 가상 머신들(즉, 하이퍼바이저와 보안 게스트 사이의)에 제공되거나 또는 다른 경계(another boundary), 예를 들어, 주소-공간의 변경을 전환의 주요 지점으로 사용하는 가상 실행가능물들(virtual executables)에 제공될 수 있다. 여기에 가상 머신 인터페이스가 설명되어 있지만 유사한 보안 인터페이스 컨트롤 인터페이스가 유사한 하드웨어 보호 메커니즘들을 사용하여 보안 실행가능물들(secure executables)제공될 수 있다.

[0030] [0051] 한 예에서, 보안 인터페이스 컨트롤은 내부의 보안이 되고 신뢰할 수 있는 하드웨어 및/또는 펌웨어에서 구현된다. 보안 게스트 또는 주체를 위해, 보안 인터페이스 컨트롤은 보안 환경의 초기화 및 유지 관리는 물론 하드웨어에 이러한 보안 주체들을 디스패치하는 것의 조정을 제공한다. 보안 게스트는 데이터를 능동적으로 사용하고 호스트 스토리지에 상주하지만 보안 스토리지에 "명확하게"로 보관된다(it is kept "in the clear" in secure storage). 보안 게스트 스토리지는 상기 단일의 보안 게스트에 의해서 액세스될 수 있는데, 이는 하드웨어에 의해 엄격하게 실행된다. 즉, 하드웨어는 모든 비-보안 주체(하이퍼바이저 또는 기타 비-보안 게스트들 포함) 또는 다른 보안 게스트들이 상기 데이터에 액세스하는 것을 금지한다. 이 예에서, 보안 인터페이스 컨트롤은 펌웨어의 가장 낮은 레벨의 신뢰할 수 있는 부분으로서 실행된다. 가장 낮은 레벨, 또는 밀리코드는 실제로 하드웨어의 확장이며, 예를 들어, IBM으로부터의 zAarchitecture®에 정의된 복잡한 명령 및 기능을 구현하는데 사용된다. 밀리코드는 보안 실행의 맥락에서 자기 자신의 보안 UV 스토리지, 비-보안 하이퍼바이저 스토리지, 보안 게스트 스토리지 및 공유 스토리지를 포함하는, 스토리지의 모든 부분들에 액세스할 수 있다. 이 것은 보안 게스트 또는 상기 게스트를 지원하는 하이퍼바이저에 의해서 필요한 모든 기능을 제공할 수 있게 해 준다. 보안 인터페이스 컨트롤은 또한 하드웨어에 직접 액세스할 수 있으므로 보안 인터페이스 컨트롤에 의해 설정된 조건들의 컨트롤 하에 하드웨어가 효율적으로 보안 체크들을 제공할 수 있게 해준다.

[0031] [0052] 본 발명의 하나 또는 그 이상의 실시 예들에 따르면, 보안 페이지를 마크(mark)하기 위해 보안-스토리지 비트(a secure-storage bit)가 하드웨어에 제공된다. 이 비트가 세트 될 때, 하드웨어는 모든 비-보안 게스트 또는 하이퍼바이저가 이 페이지에 액세스하는 것을 금지한다. 또한, 각각의 보안 또는 공유 페이지(each secure or shared page)는 존-보안 테이블에 등록되고 보안-게스트-도메인 식별(ID)로 태그 된다. 상기 페이지가 비-보안일 때, 상기 페이지는 비-보안으로 존-보안 테이블에 마크 된다. 이 존-보안 테이블은 파티션 당 또는 존 당으로(per partition or zone) 보안 인터페이스 컨트롤에 의해서 유지 관리된다. 호스트 절대 페이지당 하나의 엔트리(one entry)가 있고, 이 엔트리는 보안 주체에 의해서 만들어진 모든 DAT 변환에 관해 하드웨어에 의해서 사용되는데, 이는 상기 페이지가 그 것을 소유하는 보안 게스트 또는 주체에 의해서만 액세스되는 것을 확인하기 위해서이다. 보안 인터페이스 컨트롤 스토리지는 또한 모든(any) 비-보안 주체에 의한 액세스를 금지하기 위해 보안 페이지로도 마크 된다. 또한, 보안 인터페이스 컨트롤 스토리지는 자신 소유의 보안-도메인 ID 또는 ID들을 갖는데, 이 것(들)은 모든 다른 보안 주체가 자신의 스토리지에 액세스하는 것을 금지하기 위해 사용된다. 상기 UV 스토리지가 특정 보안 게스트 도메인과 연관될 때, 그 것은 상기 보안 게스트 도메인 ID로 또한 태그 될 수 있는데, 이는 보안 인터페이스 컨트롤 스토리지 내에서 추가 격리를 제공하기 위함이다. 이 태깅 기술은 보안 인터페이스 컨트롤 내에서 추가 보안을 허용한다. 이론상(by definition) 보안 인터페이스 컨트롤은 모든 스토리지에 액세스할 수 있지만 실제로는(by design) 보안 인터페이스 컨트롤은 이 스토리지의 각 부분(보안 게스트 스토리지 뿐 만 아니라 보안 인터페이스 컨트롤의 별도 부분들)에 신중하게 액세스하는데, 이는 하드웨어가 이들 액세스들에 관하여 보안 체크들을 시행할 수 있게 하기 위함이다.

[0032] [0053] 본 발명의 하나 또는 그 이상의 실시 예들에 따르면, 보안 인터페이스 컨트롤은 보안 인터페이스 컨트롤 자체에 의해서만 액세스될 수 있는 자기 소유의 보안 UV 스토리지를 갖는다. 이 스토리지는 보안 인터페이스 컨트롤 및 하드웨어에서 보안 게스트들에 필요한 보안을 제공하는 데 사용된다. 보안 인터페이스 컨트롤은 이 보안 스토리지를 사용하여 자신에 관한 정보, 보안 게스트들을 실행할 수 있는 존에 관한 정보, 보안 게스트들 및 보안 가상 CPU들에 대한 정보를 저장한다. 보안 게스트 스토리지와 유사한, 보안 인터페이스 컨트롤 스토리지도, 비-보안 주체의 액세스를 방지하기 위해 보안 페이지로 마크 된다. 또한, 보안 인터페이스 컨트롤 스토리지는 다른 보안 주체가 보안 인터페이스 컨트롤 스토리지에 액세스하는 것을 금지하는 데 사용되는 자신 소유의 보안 도메인 ID들을 갖는다.

[0033] [0054] 본 발명의 하나 또는 그 이상의 실시예들에 따르면, 소프트웨어는 UV 호출(UVC) 명령을 사용하여 특정 동작을 수행하기 위해 보안 인터페이스 컨트롤을 요청한다. 예를 들어, UVC 명령은 보안 인터페이스 컨트롤을 초기화하고, 보안 게스트 도메인(예: 보안 게스트 구성)을 생성하며, 상기 보안 구성 내에서 가상 CPU들을 생성하기 위해 하이퍼바이저에 의해서 사용될 수 있다. 또한, UVC 명령은 하이퍼바이저의 페이지-인 또는 페이지-아

웃 연산들의 일부로서 보안 게스트 페이지를 임포트(import)하고(해독하여 보안 게스트 도메인으로 할당함)엑스포트(export)하기(암호화하여 호스트 액세스를 허용함)위해 또한 사용될 수 있다. 또한, 보안 게스트는 하이퍼바이저와 공유되는 스토리지를 정의하고, 보안 스토리지를 공유되게 하고, 공유 스토리지를 보안이 되게 하는 능력이 있다.

[0034] [0055] 이들 UVC 명령들은 많은 다른 아키텍처 된 명령들과 유사하게 머신 펌웨어에 의해서 실행될 수 있다. 상기 머신은 보안 인터페이스 컨트롤 모드로 들어가지 않고 대신 상기 머신이 현재 실행 중인 모드에서 상기 머신은 보안 인터페이스 컨트롤 기능들을 수행한다. 하드웨어는 펌웨어와 소프트웨어 상태를 모두 유지하므로 이들 연산들을 처리하기 위해서 컨텍스트들의 스위치는 없다. 이 낮은 오버헤드는 필요한 레벨의 보안을 제공하면서 보안 인터페이스 컨트롤에서 복잡성을 최소화하고 줄이는 방식으로 소프트웨어, 신뢰할 수 있는 펌웨어 및 하드웨어의 서로 다른 계층 간의 긴밀한 협력을 허용한다.

[0035] [0056] 본 발명의 하나 또는 그 이상의 실시 예들에 따르면, 보안 게스트 및 지원 하이퍼바이저 환경들을 적절하게 유지하기 위해 보안 인터페이스 컨트롤 및 하드웨어에 의해서 필요한 컨트롤 블록 구조들의 지원으로, 하이퍼바이저는 보안 게스트 환경을 초기화하는 동안 보안 인터페이스 컨트롤에 스토리지를 기부한다(donate). 그 결과, 1) 보안 게스트들을 실행하기 위한 존 초기화(initializing a zone), 2) 보안 게스트 도메인들의 생성, 그리고 3) 상기 도메인들 각각에서 실행하는 보안 CPU들의 생성을 위한 준비로서, 하이퍼바이저는 무엇보다도 기부를 위해 필요한 스토리지 양을 결정하기 위해 쿼리 UVC 명령을 발행한다. 일단 스토리지가 기부되면, 상기 스토리지는 보안으로 마크되고 보안 인터페이스 컨트롤에 속하는 것으로 등록되고; 그리고 모든 비-보안 또는 보안 게스트 주체에 의한 액세스가 금지된다. 이는 연관된 주체(예: 보안 게스트 CPU, 보안 게스트 도메인 또는 존)가 파괴되는 때까지 유지된다.

[0036] [0057] 일 예로서, UV 스토리지의 제1 섹션(the first section of UV storage)은, 존-특정 UV 컨트롤 블록들(the zone-specific UV control blocks)을 지원하기 위해, UVC 초기화의 일부로서(as part of the initialize UVC) 보안 인터페이스 컨트롤에 기부되고 여기에서 UV2 스토리지라고 하는 곳에 상주한다(reside). UV 스토리지의 제2 및 제3 섹션은, 베이스 및 가변 보안-게스트-구성-컨트롤 블록들(the base and variable secure-guest-configuration control blocks) (각각의 보안 게스트 도메인을 위한)을 지원하기 위해, 보안-게스트-구성-생성 UVC의 일부로서(as part of the create-secure-guest-configuration UVC) 기부되고 UVS 및 UVV 스토리지에, 각각, 상주한다. UV 스토리지의 제4의 마지막 섹션은, 보안-CPU 컨트롤 블록들(the secure-CPU control blocks)을 지원하기 위해, 또한 UVS 공간에 상주하고 보안-게스트-CPU-생성 UVC의 일부로서(as part of the create-secure-guest-CPU UVC) 기부된다. 이들 영역들의 각각이 기부됨에 따라, 보안 컨트롤 인터페이스는 그들을 보안으로 마크하고(모든 비-보안 주체에 의해 그들이 액세스되는 것을 금지하기 위해) 또한 보안 컨트롤 인터페이스에 속하는 것으로 존-보안 테이블에 그들을 등록한다(모든 비-보안 주체에 의해 그들이 액세스되는 것을 금지하기 위해). UV 공간 내에서 추가 격리를 제공하기 위해, UV2 공간(모든 특정 보안-게스트 도메인과 연관되지 않은)도 또한 고유한 UV2 보안 도메인(예: 고유-보안 도메인)으로 태그 되고 한편으로 UVS 및 UVV 공간도 모두 연관된 특정 보안-게스트 도메인으로 추가 태그 된다. 이 예에서, UVV 공간은 호스트 가상 공간에 상주하고, 그러므로, 호스트 가상에서 호스트 절대로의 매핑으로(with a host virtual to host absolute mapping) 추가로 식별될 수 있다.

[0037] [0058] 보안 인터페이스 컨트롤이 모든 스토리지(비-보안 스토리지, 보안 게스트 스토리지 및 UV 스토리지)에 대한 액세스를 갖고 있기는 하지만, 본 발명의 하나 또는 그 이상의 실시 예들은 보안 인터페이스 컨트롤이 UV 스토리지에 매우 구체적으로(very specifically)액세스할 수 있도록 하는 메커니즘들을 제공한다. 보안 게스트 도메인들 간의 격리를 제공하는 동일한 하드웨어 메커니즘들을 사용하여, 본 발명의 실시 예들은 UV 스토리지 내에서 유사한 격리를 제공할 수 있다. 이 것은 다음을 보장한다: 보안 컨트롤 인터페이스는 의도되고 명시된 때(when intended and specified)만 UV 스토리지를 액세스한다; 명시된 보안 게스트가 원할 때만 보안 게스트 스토리지를 액세스한다; 그리고 명시될 때만 비-보안 스토리지를 액세스한다. 즉, 보안 컨트롤 인터페이스는 액세스하려는 스토리지를 매우 명시적으로(very explicitly) 명시할 수 있고 그렇게 하여 하드웨어는 보안 컨트롤 인터페이스가 실제로 상기 스토리지에 액세스하는 것을 보장할 수 있다. 또한, 보안 컨트롤 인터페이스는 명시된 보안 게스트 도메인과 연관된 UV 스토리지만 액세스하기를 의도한다는 것을 추가로 명시할 수 있다.

[0038] [0059] 보안을 제공하기 위해, 하이퍼바이저가 보안 게스트 데이터를 투명하게 페이지-인하고 페이지-아웃 할 때, 하드웨어와 함께 일하는, 보안 인터페이스 컨트롤은 데이터의 해독 및 암호화(the decryption and encryption of the data)를 제공하고 보장한다. 이를 달성하기 위해, UV는, 게스트 보안 데이터를 페이지-인 및 페이지-아웃 할 때, 새로운 UVC들을 발행해야 한다. 하드웨어는, 이들 새로운 UVC들 동안 보안 인터페이스 컨트

롤에 의해서 셋업 된 컨트롤들에 기초하여, 이들 UVC들이 하이퍼바이저에 의해 실제로 발행되었음을 보장할 것이다.

[0039] [0060] 이 새로운 보안 환경에서는, 하이퍼바이저가 보안 페이지를 페이지-아웃할 때마다, 보안 스토리지 (익스포트) UVC(secure storage (export) UVC)로부터 새로운 변환(new convert)을 발행하는 것이 요구된다. UV 또는 보안 인터페이스 컨트롤은, 익스포트 UVC에 응답하여, 1) 상기 페이지가 UV에 의해 "잠겨 있음(locked)"을 표시하고, 2) 상기 페이지를 암호화하며, 3) 상기 페이지를 비-보안으로 세트 하고, 4) UV 잠금을 리세트 한다. 익스포트UVC 가 완료되면, 하이퍼바이저는 이제 암호화된 게스트 페이지를 페이지-아웃할 수 있다.

[0040] [0061] 또한, 하이퍼바이저가 보안 페이지를 페이지-인 할 때마다, 하이퍼바이저는 새로운 변환을 보안 스토리지(임포트) UVC(secure storage (import) UVC)에 발행해야 한다. UV 또는 보안 인터페이스 컨트롤은, 이 임포트 UVC에 응답하여, 1) 하드웨어에서 상기 페이지를 보안으로 마크하고, 2) 상기 페이지가 UV에 의해서 "잠겼음"을 표시하며, 3) 상기 페이지를 암호 해독하고, 4) 특정 보안 게스트 도메인에 대하여 권한을 세트 하며, 그리고 5) UV 잠금을 리세트 한다. 보안 주체에 의해서 액세스될 때마다, 하드웨어는 변환 동안 상기 페이지에 관한 승인 체크들(authorization checks)을 수행한다. 이들 추가 보안 체크들에는 1) 상기 페이지가 그 것을 액세스하려고 시도하는 보안 게스트 도메인에 정말로 속하는지 확인하기 위한 체크와 그리고 2) 이 페이지가 게스트 메모리에 상주하는 동안 하이퍼바이저가 이 페이지의 호스트 매핑을 변경하지 않았음을 확인하기 위한 체크가 포함된다. 일단 페이지가 보안으로 마크 되면, 하드웨어는 모든 보안 페이지에 대한 액세스를 금지하는데, 그 것이 하이퍼바이저에 의한 액세스이던지 또는 비-보안 게스트 VM에 의한 액세스이던지 간에, 어느 것에 의한 액세스도 금지한다. 추가 변환 단계들이 다른 보안 VM에 의한 액세스도 금지하고 하이퍼바이저에 의한 재-매핑도 금지한다.

[0041] [0062] 본 발명의 하나 또는 그 이상의 실시 예들은 하이퍼바이저가 오류들과 함께 수행하고, 이에 의해서 비-보안 주체들이 보안으로 되어 있는 페이지를 액세스하는 것을 고려하지 않는 기존 시스템보다 기술적 개선을 제공한다.

[0042] [0063] 즉, 본 발명의 하나 또는 그 이상의 실시 예들은 그러한 기존 시스템의 문제를 해결하는데, 스토리지 보안 메커니즘 이외의 모든 주체에 의한 액세스를 금지하기 위해 스토리지를 보안 인터페이스 컨트롤 메모리로 그리고 보안으로, 태깅 메커니즘에 의해서, 마크하고, 그리고 상기 보안 인터페이스 컨트롤 메모리를, 보안 게스트 간의 격리를 위해 하드웨어에 제공된 스토리지 보안 메커니즘에 의해, 격리하는 것에 의해서 해결한다. 이와 관련하여, 스토리지 보안 메커니즘은 보안 인터페이스 컨트롤 메모리를 포함하는 보안 인터페이스 컨트롤이며, 보안 인터페이스 컨트롤 메모리는 상기 보안 인터페이스 컨트롤 자체에 의해서만 액세스 가능하고 그리고 보안 게스트들에 보안을 제공하기 위해 보안 인터페이스 컨트롤 및 하드웨어에 의해서 사용된다. 또한, 보안 인터페이스 컨트롤 메모리는 보안 페이지로 마크 된 보안 인터페이스 컨트롤 스토리지이고, 상기 보안 페이지는 모든 비-보안 주체에 의한 액세스를 금지하는 기술적 효과들 및 잇점들을 제공한다. 또한 보안 인터페이스 컨트롤 스토리지는 모든 다른 보안 또는 비-보안 주체들이 자신의 스토리지에 액세스 하는 것을 금지하는 데 사용되는 자기 소유의 보안 도메인 ID 또는 ID들을 갖는다. UV 스토리지가 특정 보안 게스트 도메인과 연관될 때, UV 스토리지는 상기 보안 게스트 도메인 ID로 태그 될 수 있고, 이는 보안 인터페이스 컨트롤 스토리지 내에서 추가 격리를 제공한다. 이 태깅 기술은 보안 인터페이스 컨트롤 내에서 추가 보안을 허용한다. 이론상(by definition) 보안 인터페이스 컨트롤은 모든 스토리지에 액세스할 수 있지만 실제로는(by design) 보안 인터페이스 컨트롤은 이 스토리지의 각 부분(보안 게스트 스토리지 뿐 만 아니라 보안 인터페이스 컨트롤의 별도 부분들)에 신중하게 액세스하는데, 이는 하드웨어가 이들 액세스들에 관하여 보안 체크들을 시행할 수 있게 하기 위함이다.

[0043] [0064] 이제 도 1을 참조하면, 존-보안을 위한 테이블(100)이 본 발명의 하나 또는 그 이상의 실시예들에 따라 일반적으로 도시된다. 도 1에 도시된 존-보안 테이블(100)은 보안 인터페이스 컨트롤에 의해 유지 관리되며 보안 주체가 액세스하는 모든 페이지에 대한 보안 액세스를 보장하기 위해 보안 인터페이스 컨트롤 및 하드웨어에 의해 사용된다. 존-보안 테이블(100)은 호스트 절대 주소(110)에 의해 인덱스 된다. 다시 말하면, 호스트 절대 스토리지의 각 페이지에는 하나의 엔트리(one entry) 있고, 각 엔트리는 액세스하는 보안 주체에 속하는 엔트리를 확인하는 데 사용되는 정보를 포함한다.

[0044] [0065] 또한, 도 1에 도시된 바와 같이, 존-보안 테이블(100)은 다음을 포함한다: 보안 도메인 ID(120)(이 페이지와 연관된 보안 도메인을 식별함); UV 비트(130)(이 페이지가 보안 인터페이스 컨트롤에 기부되었으며(donated) 상기 보안 인터페이스 컨트롤에 의해 소유됨을 표시함); 디스에이블 주소 비교(DA)-비트 (140)(호스

트 절대 페이지로 정의된 보안 인터페이스 컨트롤 페이지가 연관된 호스트 가상 주소를 갖지 않았을 때와 같은 특정 상황들에서 호스트 주소 쌍 비교(the host address pair compare)를 디스에이블 하는 데 사용됨); 공유(SH) 비트 (150)(페이지가 비-보안 하이퍼바이저와 공유됨을 표시함) 및 호스트 가상 주소 (160)(호스트-주소 쌍이라 하는, 이 호스트 절대 주소에 대해 등록된 호스트 가상 주소를 표시함). 호스트-주소 쌍은 호스트 절대 및 연관된, 등록된 호스트 가상 주소를 표시함을 주목해야 한다. 호스트-주소 쌍은, 일단 하이퍼바이저에 의해서 임포트된, 이 페이지의 매핑을 나타내고, 상기 비교는 상기 페이지가 게스트에 의해서 사용되고 있는 동안은 호스트가 상기 페이지를 재-매핑하지 않는다는 것을 보장한다.

[0045] [0066] 동적 주소 변환(DAT)은 가상 스토리지를 실제 스토리지에 매핑하는 데 사용된다. 게스트 VM이 하이퍼바이저의 컨트롤 하에서 페이지 가능한 게스트로 실행될 때, 상기 게스트는 DAT를 사용하여 자신의 메모리에 상주하는 페이지들을 관리한다. 또한, 호스트도, 상기 페이지들이 자신의 메모리에 상주하고 있을 때, 독립적으로, DAT를 사용하여 이들 게스트 페이지들을(자체 페이지들과 함께) 관리한다. 하이퍼바이저는 DAT를 사용하여 하이퍼바이저 스토리지에 대한 게스트 액세스를 방지할 뿐만 아니라, 서로 다른 VM 간에 스토리지의 격리 및/또는 공유를 제공한다. 하이퍼바이저는 게스트가 비-보안 모드에서 실행 중일 때 게스트들의 모든 스토리지에 액세스할 수 있다.

[0046] [0067] DAT는 애플리케이션들이 공통 자원들을 공유할 수 있도록 하면서 다른 애플리케이션으로부터 한 애플리케이션의 격리(isolation)를 인에이블 한다. 또한, DAT는 애플리케이션 프로그램들의 동시 처리와 함께 새로운 버전의 OS들의 설계 및 테스트에 사용될 수 있는, VM들의 구현을 허용한다. 가상 주소는 가상 스토리지에서의 위치를 식별한다. 주소 공간은 각 가상 주소가 스토리지의 바이트 위치를 갖는 상기 주소를 식별하는 연관된 절대 주소로 변환될 수 있도록 하는 특정 변환 파라미터들(DAT 테이블 포함)을 함께 갖는, 가상 주소의 연속적인 시퀀스(a consecutive sequence)이다.

[0047] [0068] DAT는 멀티-테이블 룩업(a multi-table lookup)을 사용하여 가상 주소를 연관된 절대 주소로 변환한다. 이 테이블 구조는 일반적으로 스토리지 관리자에 의해 정의되고 유지 관리된다. 이 스토리지 관리자는, 예를 들어, 다른 페이지를 불러올 때, 한 페이지를 페이지-아웃함으로써, 다수의 프로그램들 간에 절대 스토리지를 투명하게 공유한다. 예를 들어, 페이지가 페이지-아웃 될 때, 스토리지 관리자는 연관된 페이지 테이블에 무효 비트(invalid)를 세트 한다. 프로그램이 페이지-아웃 된 페이지에 액세스하려고 시도할 때, 하드웨어는 종종 페이지 폴트(a page fault)라고 하는 프로그램 인터럽션을 스토리지 관리자에게 제공한다. 이에 대한 응답으로, 스토리지 관리자는 요청된 페이지를 페이지-인하고 무효 비트를 리셋 한다. 이 모든 것은 프로그램에 투명하게 수행되며 스토리지 관리자가 스토리지를 가상화하고 다양한 다른 사용자들 간에 공유할 수 있게 해 준다.

[0048] [0069] 메인 스토리지에 액세스하기 위해 가상 주소가 CPU에 의해서 사용될 때, 가상주소는, 먼저 DAT의 수단에 의해서, 실제 주소로 변환되고, 그 다음 프리픽싱(prefixing)의 수단에 의해서, 절대 주소로 변환된다. 특정 주소 공간에 대한 최상위-레벨의 테이블의 지정(원점 및 길이)을 주소-공간-컨트롤 엘리멘트(an address-space-control element: ASCE)라고 하며 연관된 주소 공간을 정의한다.

[0049] [0070] 이제 도2를 참조하면, DAT를 수행하기 위한 예시적인 가상 주소 공간들(202, 204) 및 절대 주소 공간(206)이 본 발명의 하나 또는 그 이상의 실시예들에 따라 일반적으로 도시된다. 도 2에 도시된 예에서, 2개의 가상 주소 공간들이 존재한다: 가상 주소 공간(202)(주소 공간 컨트롤 엘리멘트(ASCE) A(208)에 의해 정의됨) 및 가상 주소 공간(204)(ASCE B(210)에 의해 정의됨). 가상 페이지들 A1.V(212a1), A2.V(212a2) 및 A3.V(212a3)는, 절대 페이지들A1.A (220a1), A2.A(220a2) 및 A3.A(220a3)로 매핑되는데, ASCE A(208)를 사용하여 멀티-테이블(세그먼트 230 및 페이지 테이블들 232a, 232b) 룩업에서 스토리지 관리자에 의해 매핑된다. 유사하게, 가상 페이지들 B1.V(214b1) 및 B2.V(214b2)도 절대 페이지들 B1.A(222b1) 및 B2.A(222b2)로, 각각, 매핑되는데, ASCE B(210)을 사용하여, 2개-테이블(234 및 236) 룩업에서, 매핑된다.

[0050] [0071] 이제 도 3을 참조하면, 하이퍼바이저 하에서 실행되는 VM을 지원하기 위해 사용되는 네스트된, 멀티-파트 DAT 변환(a nested, multi-part DAT translation)의 예가 본 발명의 하나 또는 그 이상의 실시예들에 따라 일반적으로 도시된다. 도 3에 도시된 예에서, 게스트 A 가상 주소 공간 A(302)(게스트 ASCE(GASCE) A(304)에 의해 정의됨) 및 게스트 B 가상 주소 공간 B(306)(GASCEB(308)에 의해 정의됨)는 모두 공유 호스트(하이퍼바이저) 가상 주소 공간에 상주한다. 도시된 바와 같이, 게스트 A에 속하는 가상 페이지 A1.GV(310a1), A2.GV(310a2), 및 A3.GV(310a3)는, 게스트 절대 페이지 A1.HV(340a1), A2.HV(340a2) 및 A3.HV (340a3)로, 각각, 매핑 되는데, GASCEA(304)를 사용하여, 게스트 A 스토리지 관리자에 의해 매핑 된다; 게스트 B에 속하는 가상 페이지 B1.GV(320b1) 및 B2.GV(320b1)는, 게스트 절대 페이지 B1.HV(360b1) 및 B2.HV(360b2)에, 각각, 매핑 되는데,

GASCEB(308)을 사용하여, 게스트 B 스토리지 관리자에 의해 매핑된다. 이 예에서, 이들 게스트 절대 페이지들은 공유 호스트 가상 주소 공간에 직접 매핑되고 후속적으로 추가 호스트 DAT 변환을 통해 호스트 절대 주소 공간(330)으로 나아간다. 도시된 바와 같이 호스트 가상 주소들 A1.HV (340a1), A3.HV(340a3), 및 B1.HV(360b1)은, A1.HA(370a1), A3.HA(370a3) 및 B1.HA(370b1)에 매핑 되는데, 호스트 ASCE(HASCE)(350)을 사용하여 호스트 스토리지 관리자에 의해 매핑된다. 게스트 A에 속하는 호스트 가상 주소 A2.HV (340a2)와 게스트 B에 속하는 B2.HV(360b2)는 모두 동일한 호스트 절대 페이지 AB2.HA(380)에 매핑 된다. 이것은 두 개의 게스트들 간에 데이터를 공유될 수 있게 한다. 게스트 DAT 변환 동안에 게스트 테이블 주소들 각각은 게스트 절대 주소로 취급되어 추가의 네스트된 호스트 DAT 변환을 거친다(undergo).

[0051] [0072] 여기서 설명된 본 발명의 실시예들은 보안 게스트 및 UV 스토리지 보호를 제공한다. 비-보안 게스트들 및 하이퍼바이저에 의한 보안 스토리지 액세스는 금지된다. 하이퍼바이저는, 주어진 상주 보안 게스트 페이지에 대해, 다음에서 발생하는 것을 제공한다. 연관된 호스트 절대 주소는 단일 하이퍼바이저(호스트) DAT 매핑을 통해서만 액세스가 가능하다. 즉, 보안 게스트에 할당된 모든 주어진 호스트 절대 주소로 매핑되는 단일 호스트 가상 주소가 있다. 주어진 보안 게스트 페이지와 연관된 하이퍼바이저 DAT 매핑(호스트 가상에서 호스트 절대값으로)은 상기 페이지가 페이지-인 되는 동안은 변경되지 않는다. 보안 게스트 페이지와 연관된 호스트 절대 페이지는 단일 보안 게스트에 대해 매핑 된다.

[0052] [0073] 보안 게스트들 간의 스토리지 공유도 본 발명의 하나 또는 그 이상의 실시예들에 따라 또한 금지된다. 스토리지는 단일 보안 게스트와 상기 보안 게스트의 컨트롤 하에 있는 하이퍼바이저 간에 공유된다. UV 스토리지는 보안 스토리지이며 보안 인터페이스 컨트롤에 의해서 액세스 가능하지만 게스트들/호스트들에 의해서는 액세스 가능하지 않다. 스토리지는 하이퍼바이저에 의해서 보안 인터페이스 컨트롤에 할당된다. 본 발명의 하나 또는 그 이상의 실시예들에 따르면, 이러한 규칙의 위반 시도는 하드웨어 및 보안 인터페이스 컨트롤에 의해 금지된다.

[0053] [0074] 이제 도 4를 참조하면, 보안 게스트 스토리지의 매핑의 한 예가 본 발명의 하나 또는 그 이상의 실시예들에 따라 일반적으로 도시되어 있다. 도 4는 도 3와 유사하다. 도 4의 예는 보안 게스트 A와 보안 게스트 B 간의 스토리지 공유를 허용하지 않는다는 점을 제외하고는 도 3과 유사하다. 도 3의 비-보안 예에서, 게스트 A에 속하는 호스트 가상 주소 A2.HV(340a2)와 게스트 B에 속하는 B2.HV(360b2) 모두는 동일한 호스트 절대 페이지 AB2.HA(380)에 매핑된다. 도 4의 보안 게스트 스토리지 예에서, 게스트 A에 속하는 호스트 가상 주소 A2.HV(340a2)는 호스트 절대 주소 A2.HA (490a)에 매핑 되는 반면 게스트 B에 속하는 B2.HV(360b2)는 자신의 B2.HA(490b)에 매핑 된다. 이 예에서, 보안 게스트들 간에는 공유가 없다.

[0054] [0075] 보안 게스트 페이지는 디스크에 상주하는 동안, 그것은 암호화된다. 하이퍼바이저가 보안 게스트 페이지에서 페이지-인 할 때, 하이퍼바이저는 UV 호출(UVC)을 발행하는데, 이 호출은 보안 컨트롤 인터페이스로 하여금 상기 페이지를 보안으로 마크하게 하고(공유되지 않는 한), 상기 페이지를 해독(decrypt)하게 하며(공유되지 않는 한), 그리고 적절한 보안 게스트(예: 게스트 A)에 속하는 것으로서 등록하게 한다(존 보안 테이블 안에).또한, 하이퍼바이저는 연관된 호스트 가상 주소(예: A3.HV 340a3)를 상기 호스트 절대 페이지(호스트-주소 쌍이라고 함)에 등록한다. 만일 하이퍼바이저가 올바른(correct) UVC를 발행하는 데 실패하면, 하이퍼바이저는 보안 게스트 페이지에 액세스하려고 시도할 때 예외를 수신한다. 하이퍼바이저가 게스트 페이지를 페이지-아웃 할 때, 게스트 페이지를 비-보안으로 표시하고 그 것을 존-보안 테이블에 비-보안으로 등록하기 전에 게스트 페이지를 암호화하는(공유되지 않은 한) 유사한 UVC가 발행된다.

[0055] [0076] 주어진 5개의 호스트 절대 페이지 K, P, L, M 및 N을 갖는 예에서, 호스트 절대 페이지들의 각각은 하이퍼바이저가 상기 호스트 절대 페이지들을 페이지-인 할 때 보안 컨트롤 인터페이스에 의해서 보안으로 마크된다. 이 것은 비-보안 게스트들과 하이퍼바이저가 상기 호스트 절대 페이지들을 액세스 하는 것을 금지한다. 호스트 절대 페이지들 K, P 및 M은 하이퍼바이저가 그들을 페이지-인 할 때 게스트 A에 속하는 것으로 등록되며; 호스트 절대 페이지들 L 및 N은 하이퍼바이저에 의해 페이지-인 될 때 게스트 B에 등록된다. 단일 보안 게스트와 하이퍼바이저 간에 공유되는 페이지들인, 공유 페이지들은 페이징 동안에 암호화되거나 해독되지 않는다. 그들은 보안으로 마크 되지는 않지만(하이퍼바이저에 의한 액세스 허용) 존 보안 테이블의 단일 보안 게스트 도메인에 등록된다.

[0056] [0077] 본 발명의 하나 또는 그 이상의 실시예들에 따르면, 비-보안 게스트 또는 하이퍼바이저가 보안 게스트에 의해서 소유된 페이지를 액세스하려고 시도할 때, 하이퍼바이저는 보안 스토리지 액세스(PIC3D) 예외를 수신한다. 이를 결정하기 위해 추가 변환 단계가 필요하지 않다.

[0057] [0078] 하나 또는 그 이상의 실시예들에 따라, 보안 주체가 하나의 페이지에 액세스하려고 시도할 때, 하드웨어는 스토리지가 실제로 그 특정 보안 게스트에 속한다는 것을 확인하기 위한 추가의 변환 체크(an additional translation check)를 수행한다. 만일 그렇게 하지 않는다면, 비-보안 액세스(PIC3E) 예외가 하이퍼바이저에 제공된다. 또한, 만일 변환되는 호스트 가상 주소가 존 보안 테이블에 등록된 호스트-주소 쌍으로부터의 호스트 가상 주소와 매치하지 않는다면, 보안-스토리지 위반('3F'x) 예외가 인지된다. 하이퍼바이저와 공유하는 것을 인에이블 하기 위해, 보안 게스트는 변환 체크들이 액세스를 허용하는 한 보안으로 마크되지 않은 스토리지에 액세스할 수 있다.

[0058] [0079] 이제 도 5로 돌아가면, DAT 연산의 시스템 개략도(500)가 일반적으로 본 발명의 하나 또는 그 이상의 실시 예들에 따라 도시된다. 시스템 개략도(500)는 호스트 초기 가상 주소 공간(a host primary virtual address space)(510)와 호스트 홈 가상 주소 공간(a host home virtual address space)(520)를 포함하고, 이들로부터 페이지들이 하이퍼바이저(호스트) 절대 주소 공간(530)으로 변환된다(예를 들어, 호스트 DAT 변환(525)을 참조한다; 점선들은 DAT변환(525)을 통한 매핑을 나타낸다는 것에 유의한다). 예를 들어, 도 5는 두 개의 서로 다른 호스트 가상 주소 공간들에 의해서 호스트 절대 스토리지를 공유하는 것(sharing)과 또한 두 개의 게스트들 사이에서뿐만아니라, 추가로 호스트 자체와 이들 호스트 가상 주소들 중 하나를 공유하는 것을 예시한다. 이와 관련하여, 호스트 초기 가상 주소 공간(510)와 호스트 홈 가상 주소 공간(520)는 두 호스트 가상 주소 공간들의 예들이고, 이들 각각은, 별개의 ASCE, 호스트 초기 ASCE(HPASCE)(591), 및 호스트 홈 ASCE(HHASCE)(592)에 의해서 어드레스 된다(addressed). 모든 보안 인터페이스 컨트롤 스토리지(가상 및 실제 모두)는 하이퍼바이저에 의해서 기증되고(donated) 보안으로 마크된다(mark)된다는 것에 유의한다. 일단 기증되면, 보안 인터페이스 컨트롤 스토리지는 연관된 보안 주체가 존재하는 동안에 한해서 오직 보안 인터페이스 컨트롤에 의해서만 액세스될 수 있다.

[0059] [0080] 예시된 바와 같이, 호스트 초기 가상 주소 공간(510)는 게스트 A 절대 페이지 A1.HV, 게스트 A 절대 페이지 A2.HV, 게스트 B 절대 페이지 B1.HV 및 호스트 가상 페이지 H3.HV를 포함한다. 호스트 홈 가상 주소 페이지(520)는 보안-인터페이스-컨트롤(secure-interface-control) 가상 페이지 U1.HV, 호스트 가상 페이지 H1.HV 및 호스트 가상 페이지 H2.HV를 포함한다.

[0060] [0081] 본 발명의 하나 또는 그 이상의 실시 예들에 따라, 모든 보안 게스트(예: 보안 게스트 A & 보안 게스트 B) 스토리지는, 보안 게스트 구성에 속하는 것으로, 여기에 설명된 존-보안 테이블(zone-security table)에서, 등록되고, 연관된 호스트 가상 주소(예: A1.HV, A2.HV, B1.HV)도 또한 호스트-주소 쌍의 일부로서(as a part of host-address pair) 등록된다. 하나 또는 그 이상의 실시 예들에서, 모든 보안 게스트 스토리지가 호스트 초기 가상 공간에서 매핑 된다. 이에 더하여, 모든 보안 인터페이스 컨트롤 스토리지도 또한, 보안 인터페이스 컨트롤에 속하는 것으로, 존-보안 테이블에도 등록되고, 더 나아가서, 연관된 보안 게스트 도메인에 기초하여 존-보안 테이블에서 구별될 수 있다(differentiated). 본 발명의 하나 또는 그 이상의 실시 예들에 따라, UV 가상 스토리지는 호스트 홈 가상 주소 공간에 매핑 되고, 연관된 호스트 가상 주소는 호스트-주소 쌍의 일부로서 등록된다. 하나 또는 그 이상의 실시 예들에 따라, UV 실제 스토리지는 연관된 호스트 가상 매핑이 없고, 존-보안 테이블의 DA 비트(이는 가상 주소 비교가 디스에이블 되었음을 표시함)는 이를 표시하도록 세트(set)된다. 호스트 스토리지는 비-보안(non-secure)으로 마크 되고 또한 존-보안 테이블에도 비-보안으로 등록된다.

[0061] 0082] 따라서, '게스트 절대 = 호스트 가상'인 경우에, 하이퍼바이저(호스트) 초기 DAT 테이블들(HPASCE(591)에 의해서 정의된)은 호스트 초기 가상 주소 공간(510)의 페이지들을 다음과 같이 변환한다: 게스트 A 절대 페이지 A1.HV는 보안 게스트 A에 속하는 호스트 절대 A1.HA에 매핑 된다; 게스트 A 절대 페이지 A2.HV는 보안 게스트 A에 속하는 호스트 절대 A2.HA에 매핑 된다; 게스트 B 절대 페이지 B1.HV는 보안 게스트 B에 속하는 호스트 절대 B1.HA에 매핑 된다; 그리고 호스트 가상 페이지 H3.HV는 호스트 절대 페이지 H3.HA 비-보안 호스트 에 매핑 된다(그리고 비-보안 호스트이므로 호스트-주소 쌍이 없다). 또한, 하이퍼바이저(호스트) 홈 DAT 테이블들(HASCE(592)에 의해 정의된)은 호스트 홈 가상 주소 공간(520)의 페이지들을 다음과 같이 변환한다: 보안 인터페이스 컨트롤 가상 페이지 U1.HV는 보안UV 가상으로 정의된 호스트 절대 페이지 U1.HA에 매핑 된다; 호스트 가상 페이지 H1.HV는 비-보안으로 정의된 호스트 절대 페이지 H1.HA에 매핑 된다; 그리고 호스트 가상 페이지 H2.HV는 비-보안으로 정의된 호스트 절대 페이지 H2.HA에 매핑 된다. H1.HA나 또는 H2.HA와 연관된 호스트-주소 쌍이 없는 데, 이는 그들이 비-보안이기 때문이다.

[0062] [0083] 연산에서, 만일 보안 게스트가 보안 인터페이스 컨트롤에 할당된 보안 페이지에 액세스하기 위해 시도하면(tries), 보안-스토리지 위반('3F'X) 예외가 하드웨어에 의해서 하이퍼바이저에 제공된다(presented). 만일 비-보안 게스트 또는 하이퍼바이저가 모든 보안 페이지(보안 인터페이스 컨트롤에 할당된 페이지들을 포함)에

액세스하기 위해 시도하면, 보안-스토리지 액세스('3D'X) 예외가 하드웨어에 의해서 하이퍼바이저에 제공된다(presented). 이와 달리, 오류 상태(condition)가 보안 인터페이스 컨트롤 공간에 대해 만들어진 시도된 액세스에 대해 제공될 수 있다. 만일 하드웨어가 보안 인터페이스 컨트롤 액세스에서 보안 할당의 미스매치를 검출한다면(예를 들어, 스토리지가 보안 인터페이스 컨트롤이 아니라 보안 게스트에 속하는 것으로 존-보안 테이블에 등록되었다면, 또는 등록된 쌍으로 사용되는 호스트-주소 쌍에 미스매치가 있다면)를 검출한다면, 체크가 제공된다.

[0063] [0084] 다시 말하면, 호스트 초기 가상 주소 공간(510)은 호스트 가상 페이지들A1.HV와 A2.HV(보안 게스트 A에 속하는) 및 B1.HV(보안 게스트 B에 속하는)를 포함하며, 이들은, 각각, 호스트 절대 페이지들 A1.HA, A2.HA 및 B1.HA로 매핑 된다. 또한, 호스트 초기 가상 주소 공간(510)는 호스트(하이퍼바이저) 페이지 H3.HV를 포함하며, 이는 호스트 절대 페이지H3.HA에 매핑 된다. 호스트 홈 가상 공간(520)는 두 개의 호스트 가상 페이지들 H1.HV 및 H2.HV를 포함하며, 이들은 호스트 절대 페이지들 H1.HA 및 H2.HA로 매핑 된다. 호스트 초기 가상 주소 공간(510) 및 호스트 홈 가상 주소 공간(520) 모두는 단일 호스트 절대 주소 공간(530)에 매핑 된다. 보안 게스트 A 및 보안 게스트 B에 속하는 스토리지 페이지들은 보안으로 마크 되고 그들의 보안 도메인들 및 연관된 호스트 가상 주소들과 함께 도 1에 도시된 존-보안 테이블(100)에 등록된다. 반면에, 호스트 스토리지는 비-보안으로 마크 된다. 하이퍼바이저가 보안 게스트들을 정의할 때, 하이퍼바이저는 호스트 스토리지를 보안 인터페이스 컨트롤에 기증(donate)해야 하는데, 이는 이들 보안 게스트들의 지원에 필요한 보안 컨트롤 블록들에 사용하기 위해서이다. 이 스토리지는 호스트 절대 공간 또는 호스트 가상 공간 어느 곳에서던지 정의될 수 있고, 일 예에서, 특히 호스트 홈 가상 공간에서 정의될 수 있다. 도 5로 돌아가면, 호스트 절대 페이지들 U1.HA 및 U2.HA 보안 UV 절대(a host absolute pages U1.HA and U2.HA Secure UV Absolute)는 호스트 절대 스토리지로 정의된 보안-인터페이스-컨트롤 스토리지이다. 그 결과, 이들 페이지들은 보안으로 마크 되고 연관된 보안 도메인과 함께 보안 인터페이스 컨트롤에 속하는 것으로 도 1에 도시된 존-보안 테이블(100)에 등록된다. 상기 페이지들은 호스트 절대 주소들로 정의되기 때문에, 연관된 호스트 가상 주소는 없고 따라서 DA 비트가 존-보안 테이블(100)에서 세트(set)된다.

[0064] [0085] 변환 후, 하이퍼바이저(호스트) 절대 주소 공간(530)의 일 예는 도 6에서 찾을 수 있다. 보안 인터페이스 컨트롤 메모리에 관한 도 6의 시스템 개략도(system schematic)(600)가 본 발명의 하나 또는 그 이상의 실시예들에 따라 설명된다. 시스템 개략도(600)는 하이퍼바이저(호스트) 절대 주소 공간(630)를 예시하고, 하이퍼바이저(호스트) 절대 주소 공간(630)은 호스트 절대 페이지 A2.HA 보안 게스트 A(A2.HV를 위한); 호스트 절대 페이지 B1.HA 보안 게스트 B(B1.HV를 위한); 호스트 절대 페이지 H1.HA 비-보안(호스트); 호스트 절대 페이지 H2.HA 비-보안(호스트); 호스트 절대 페이지 U3.HA 보안 UV 실제(HV 매핑 없음); 호스트 절대 페이지 U1.HA 보안 UV 가상(U1.HV를 위한); 및 호스트 절대 페이지 A1.HA 보안 게스트 A(A1.HV를 위한)를 포함한다.

[0065] [0086] 이제 도 7를 참조하면, 하나의 임포트 연산(an import operation)을 위한 프로세스 플로(700)가 본 발명의 하나 또는 그 이상의 실시예들에 따라 일반적으로 도시되어 있다. 보안 게스트가 하이퍼바이저에 의해 페이지 아웃된 페이지에 액세스할 때 상기 페이지를 안전하게 다시 임포트 하기 위해 프로세스 플로(700)에 표시된 것과 같은 일련의 이벤트가 발생한다. 프로세스 플로(700)는 보안 게스트가 게스트 가상 페이지에 액세스하는, 블록(705)에서 시작한다. 예를 들어, 상기 페이지가 유효하지 않기 때문에, 하드웨어는, 프로그램-인터럽션-코드 11(PIC11)에 의해서 표시되는, 호스트 페이지 오류(a host page fault)를 하이퍼바이저에 제공한다(블록 715 참조). 그 다음, 하이퍼바이저는, 이 게스트 페이지를 위해 이용 가능한 비-보안 호스트 절대 페이지를 식별하고(블록 720 참조), 식별된 호스트 절대 페이지로 암호화된 게스트 페이지를 페이지-인 한다(블록 725 참조).

[0066] [0087] 블록(730)에서, 상기 호스트 절대 페이지는 적절한(호스트 가상 주소에 기초한) 호스트 DAT 테이블에 매핑 된다. 블록(735)에서, 하이퍼바이저 호스트는 보안 게스트를 재-디스패치 한다. 블록(740)에서, 보안 게스트는 게스트 보안 페이지에 재-액세스한다. 페이지 폴트는 더 이상 존재하지 않지만 이것이 보안 게스트 액세스이고 상기 페이지는 도 1의 존 보안 테이블(the zone-security table)(100)에서 보안으로 마크 되지 않기 때문에 하드웨어는 블록(745)에서 하이퍼바이저에 비-보안 스토리지 예외(PIC3E)를 제공한다. 이 PIC3E는 필요한 임포트 명령(import instruction)이 발행될 때까지 게스트가 이 보안 페이지에 액세스하는 것을 금지한다. 다음으로, 프로세스 플로(500)는 도 8에 연결된 "A"로 진행한다.

[0067] [0088] 이제 도 8을 참조하면, 임포트 연산(an import operation)을 수행하기 위한 프로세스 플로(800)가 본 발명의 하나 또는 그 이상의 실시예들에 따라 일반적으로 도시된다. 잘 작동하는 하이퍼바이저(A well-behaved hypervisor)(예: 오류 없이 예상된 방식으로 수행하는)는 PIC3E에 응답하여 임포트 UVC를 발행한다(블록 805 참

조). 이 시점에서 임포트 될 페이지는 비-보안으로 마크 되고 하이퍼바이저, 기타 비보안 주체 및 보안 인터페이스 컨트롤에 의해서만 액세스 될 수 있다. 임포트 될 페이지는 보안 게스트들에 의해서는 액세스 될 수 없다.

[0068] [0089] 임포트 UVC의 일부로서, 보안 인터페이스 컨트롤 역할을 하는 신뢰할 수 있는 펌웨어는 이 페이지가 보안 인터페이스 컨트롤에 의해 이미 잠겨 있는지를 확인한다(결정 블록 810 참조). 만약 그렇다면, 프로세스 플로(800)는 블록(820)으로 진행한다. 블록(820)에서, "비지(busy)" 리턴 코드가 하이퍼바이저로 리턴 되고, 응답으로 하이퍼바이저는 임포트UVC를 지연(블록 825 참조) 및 재발행(프로세스 플로800이 블록805으로 리턴함) 시킬 것이다. 만일 상기 페이지가 이미 잠겨 있지 않다면, 프로세스 플로(800)는 결정 블록(822)으로 진행한다.

[0069] [0090] 결정 블록(822)에서, 보안 인터페이스 컨트롤은 상기 페이지가 비-보안 하이퍼바이저와 공유되는 페이지인지 확인하기 위해 체크한다. 만일 공유되는 경우(프로세스 플로 800은 결정 블록 824로 진행), 보안 인터페이스 컨트롤은 존 보안 테이블 내의 호스트 절대 주소를 연관된 보안 게스트 도메인, 호스트 가상 주소에 공유로서(as shared) 등록한다. 이 페이지는 비-보안으로 마크된 상태로 유지된다. 이렇게 하면 UVC 임포트가 완료되고 상기 페이지는 이제 게스트에 의해 액세스가 가능하게 된다. 처리하는 단계는 하이퍼바이저의 게스트 재-디스패칭 단계(블록 830) 및 보안 게스트의 성공적인 상기 페이지 액세스 단계(블록 835)로 진행되어 계속된다.

[0070] [0091] 만일 임포트될(to be imported) 호스트 가상 페이지가 하이퍼바이저와 공유되지 않는 다면(프로세스 플로 800은 블록 840으로 진행), 보안 인터페이스 컨트롤은 하이퍼바이저가 더 이상 상기 페이지에 액세스할 수 없도록 상기 페이지를 보안으로 마크한다(mark). 블록(845)에서 보안 인터페이스 컨트롤은 상기 페이지를 잠그므로, 어떤 다른 UVC도 상기 페이지 상태를 수정할 수 없다. 잠금이 세트 되면(블록 850에서), 보안 인터페이스 컨트롤은 게스트 페이지의 컨텐츠가 암호화되는 동안 변경되지 않았는지 확인한다. 만일 변경되었다면, 오류 리턴 코드가 하이퍼바이저에 리턴 되고, 그렇지 않으면, 보안 인터페이스 컨트롤이 상기 보안 페이지의 암호를 해독한다.

[0071] [0092] 블록 855에서, 보안 인터페이스 컨트롤은 상기 페이지의 잠금을 해제하여, 다른 UVC들에 의한 액세스를 허용하며, 존 보안 테이블에 상기 페이지를 등록하는데, 호스트-주소 HV->HA 쌍을 완료하기 위해 보안으로서 그리고 적절한 게스트 도메인 및 호스트 가상 주소와 연관되게 한다. 이것은 게스트에 의한 액세스를 허용하고 상기 UVC를 완료한다.

[0072] [0093] 이제 도 9로 돌아가면, 기증된 메모리 연산에 관한 프로세스 플로(process flow)(900)이 본 발명의 하나 또는 그 이상의 실시 예들에 따라 일반적으로 도시된다. 프로세스 플로(900)는 하이퍼바이저가 보안 인터페이스 컨트롤에 쿼리- UVC(query- UVC)를 발행하는, 블록(905)에서 시작한다. 블록(910)에서, 보안 인터페이스 컨트롤은 데이터(예: 쿼리 UVC)를 리턴 한다. 이러한 데이터는 다음을 포함할 수 있다; 요구되는 베이스 존-특정 호스트-절대 스토리지의 양(an amount of base zone-specific host-absolute storage required); 요구되는 베이스 보안-게스트-도메인-특정 호스트-절대 스토리지의 양(an amount of base secure-guest-domain-specific host-absolute storage required); MB당 요구되는 가변 보안-게스트-도메인-특정 호스트-가상 스토리지의 양(an amount of variable secure-guest-domain-specific host-virtual storage required per MB); 및/또는 요구되는 베이스 보안-게스트-CPU-특정 호스트-절대 스토리지의 양(amount of base secure-guest-CPU-specific host-absolute storage required).

[0073] [0094] 블록(915)에서, 하이퍼바이저는 베이스 호스트-절대 존-특정 스토리지(예: 쿼리 UVC에 의해 리턴 된 크기에 기초하는)를 유보한다 (reserves). 블록(920)에서, 하이퍼바이저는 보안 인터페이스 컨트롤에 대해 초기화(an initialization)를 발행한다. 이와 관련하여, 하이퍼바이저는 UVC 초기화 (an initialize UVC)를 발행할 수 있는데, 이 것은 전체 존에 대한 보안 게스트 구성들 사이를 조정하는 데 필요한 UV 컨트롤 블록들을 위해 기증된 스토리지를 제공한다. 상기 초기화 UVC는 베이스 존-특정 스토리지 출처(origin)를 명시한다.

[0074] [0095] 블록(925)에서, 보안 인터페이스 컨트롤은 상기 초기화(예: UVC를 초기화(initialize))를 UV에 기증된 스토리지를 등록하고 보안으로 마크함으로써 구현한다. 상기 초기화 UVC를 위해, 보안 인터페이스 컨트롤은 기증된 스토리지를 보안으로 마크하고; 상기 기증된 스토리지 중 일부(some)를 존-보안 테이블을 위해 할당하며; 그리고 기증된 스토리지를 UV 사용을 위해 존-보안 테이블에 등록할 수 있는데, 고유한 보안-도메인과는 함께(with a unique secure-domain) 등록할 수 있지만, 연관된 보안-게스트-도메인과는 함께 등록할 없는데 연관된 호스트-가상 주소 쌍을 갖고 있지 않기 때문이다.

[0075] [0096] 블록(930)에서, 하이퍼바이저는 스토리지(예: 베이스 및 가변 보안-게스트-도메인-특정 스토리지)를 유보한다. 예를 들어, 하이퍼바이저는 베이스 및 가변(예: 보안-게스트-도메인 스토리지의 크기에 기초하는) 보

안-게스트-도메인-특정 스토리지(예: 쿼리 UVC에 의해 리턴 된 크기)를 유보한다. 블록(935)에서, 하이퍼바이저는 보안 인터페이스 컨트롤에 대해 구성 생성(create configuration)을 발행한다. 이와 관련하여, 하이퍼바이저는 베이스 및 가변 보안-게스트-도메인-특정 스토리지 출처(origin)를 명시하는 보안-게스트-구성-생성(create-secure-guest-config) UVC를 발행할 수 있다. 또한, 보안-게스트-구성-생성UVC는 이러한 보안 게스트 구성을 지원하는 데 필요한 UV 컨트롤 블록들을 위해 기증된 스토리지를 제공한다.

[0076] [0097] 블록(940)에서, 보안 인터페이스 컨트롤은 구성 생성(예: 보안-게스트-구성-생성UVC)을 구현한다. 보안-게스트-구성-생성 UVC를 위해, 보안 인터페이스 컨트롤은 기증된 스토리지를 보안으로 마크할 수 있고; 기증된 스토리지를 UV 사용을 위해 존-보안 테이블에 등록할 수 있으며; 그리고 기증된 스토리지를 연관된 보안-게스트-도메인(secure-guest-domain)에 등록할 수 있다. 기증된 베이스(호스트-절대) 스토리지는 연관된 호스트-가상 주소 쌍을 갖지 않는 것으로 등록된다. 기증된 가변(호스트-가상) 스토리지는 연관된 호스트-가상 주소 쌍에 등록된다.

[0077] [0098] 블록(945)에서, 하이퍼바이저는 베이스 보안-게스트-CPU-특정 스토리지(예: 쿼리-UV에 의해 리턴 된 크기)를 유보한다. 블록(950)에서, 하이퍼바이저는 스토리지 출처(origin)를 명시한다. 예를 들어, 하이퍼바이저는 베이스 보안-게스트-CPU-특정 스토리지 출처를 명시하는 보안-게스트-CPU-생성 UV(the UV create-secure-guest-CPU)를 발행한다. 블록(955)에서, 보안 인터페이스 컨트롤은 CPU-생성(create-CPU)(예: 보안-게스트-CPU-생성 UVC)를 구현한다. 보안-게스트-CPU-생성 UVC를 위해, 보안 인터페이스 컨트롤은 기증된 스토리지를 보안으로 마크할 수 있고, 기증된 스토리지를 UV 사용을 위해 존-보안 테이블에 등록할 수 있지만, 연관된 보안-게스트-도메인으로 등록할 수는 없는데 연관된 호스트-가상 주소 쌍을 갖지 않기 때문이다.

[0078] [0099] 이제 도 10으로 돌아가면, 보안 인터페이스 컨트롤의 보안 페이지들로 비-보안 하이퍼바이저 페이지들의 전환에 관한 프로세스 플로(1000)가 본 발명의 하나 또는 그 이상의 실시 예들에 따라 일반적으로 도시된다. 프로세스 플로(1000)에서, 3개의 하이퍼바이저 페이지들이 도시된다(예: 비-보안 하이퍼바이저 페이지 A, 비-보안 하이퍼바이저 페이지 B, 및 비-보안 하이퍼바이저 페이지 C).

[0079] [0100] 하이퍼바이저(비-보안) 페이지들 A, B 및 C는 비-보안 주체 (하이퍼바이저를 포함)에 의해서 액세스될 수 있다. 또한, 하이퍼바이저(비-보안) 페이지들 A, B 및 C는 비-보안(NS)으로 마크 되며, 이와 함께 존-보안 테이블(예: 도 1에 도시된 존-보안 테이블(100))에 비-보안 및 공유-안됨(non-shared)으로 등록된다. 화살표(arrow)(1005)에서, UVC 초기화(an initialize UVC)가 발행되며, 이는 게스트 페이지 A가 전체 존(UV2)과 연관된 인터페이스 컨트롤 실제 스토리지 페이지(1010)로 전환한다.보안 인터페이스 컨트롤 실제 스토리지(1010)는 보안으로 마크 될 수 있고, 이와 함께 보안 게스트 도메인도 없고 하이퍼바이저에서 호스트 절대로(HV->HA) 매핑도 없는(no hypervisor to host absolute (HV->HA) mapping) UV로서 존-보안 테이블(예: 도 1에 도시된 존-보안 테이블(100))에 등록될 수 있다. 대신 보안 인터페이스 컨트롤 실제 스토리지(1010)는 고유한 UV2 보안 도메인에 등록되고 DA 비트가 1로 세트 된다. 보안 인터페이스 컨트롤 실제 스토리지(1010)는 보안 인터페이스 컨트롤에 의해서 실제로 액세스될 수 있음에 유의한다.

[0080] [0101] 하이퍼바이저(비-보안) 페이지 B로부터, 화살표(1025)에서, SG-구성-생성(create-SG-config) 또는 SG-CPU-생성(create-SG-CPU) UVC가 발행되며, 이는 이 페이지를 보안 게스트 도메인(UVS)과 연관된 보안 인터페이스 컨트롤 실제 스토리지(1030)로 전환한다(transition). 보안 인터페이스 컨트롤 실제 스토리지(1030)는 보안으로 마크 될 수 있고, 이와 함께 연관된 보안 게스트 도메인은 있고 하이퍼바이저에서 호스트 절대로 (HV->HA) 매핑(no hypervisor to host absolute (HV->HA) mapping)(즉, DA-bit=1)은 없는 UV로 존-보안 테이블(예: 도 1에 도시된 존-보안 테이블(100))에 등록될 수 있다. 보안 인터페이스 컨트롤 실제 스토리지(1010)는 보안 게스트 도메인을 대신하여 보안 인터페이스 컨트롤에 의해서 실제로 액세스될 수 있다.

[0081] [0102] 하이퍼바이저(비-보안) 페이지 C로부터, 화살표(1045)에서, SG-구성-생성 UVC가 발행되며, 이는 이 페이지를 보안 게스트 도메인(UVV)과 연관된 보안 인터페이스 컨트롤 가상 스토리지(1050)로 전환한다. 보안 인터페이스 컨트롤 가상 스토리지(1050)는 보안으로 표시될 수 있고, 이와 함께 보안 게스트 도메인도 있고 하이퍼바이저에서 호스트 절대로(HV->HA) 매핑도 있는 UV로 존-보안 테이블(예: 도 1에 도시된 존-보안 테이블(100))에 등록될 수 있다. 보안 인터페이스 컨트롤 가상 스토리지(1050)은 보안 게스트 도메인을 대신하여 UV 가상으로 액세스될 수 있다.

[0082] [0103] 이제 도 11로 돌아가면, 프로그램 또는 보안 인터페이스 컨트롤에 의한 보안 스토리지 액세스에 관한 프로세스 플로(1100)가 하나 또는 그 이상의 실시 예들에 따라 도시된다. 이는 보안 인터페이스 컨트롤이 게스트 스토리지 또는 보안 인터페이스 컨트롤 스토리지에 액세스하려고 하고 하드웨어가 상기 액세스의 보안을 확

인(verify)할 수 있도록 하기 위해 상기 액세스에 올바르게 태그 해야 하는 상황을 나타낸다(represents). 프로세스 플로(1100)은 보안 인터페이스 컨트롤에 의한 스토리지 액세스들의 이러한 태깅을 설명한다. 프로세스 플로(1100)은, 보안 인터페이스 컨트롤이 보안 인터페이스 컨트롤 스토리지에 액세스하는지를 결정하는 블록(1110)에서, 시작한다(begins).

[0083] [0104] 만일 이 것이 보안 인터페이스 컨트롤 스토리지에 대한 액세스가 아니면, 프로세스 플로(1100)은 결정(decision) 블록(1112)(NO 화살표에 의해 도시됨)으로 진행한다. 결정 블록(1112)에서, 보안 인터페이스 컨트롤은 보안 게스트 스토리지에 액세스하는지를 결정한다. 만일 보안 게스트 스토리지에 액세스가 아니면, 프로세스 플로(1100)은 비-보안 액세스들에 대해 디폴트 셋팅을 사용하는 "B"(도 12의 프로세스 플로(1200)에 관련됨)로 진행한다. 만일 보안 게스트 스토리지에 대한 액세스라면, 프로세스 플로(1100)은 결정 블록(1113)으로 진행되며, 여기서 보안 인터페이스 컨트롤은 만일 다폴트 보안 게스트 도메인이 사용되고 있는지를 결정한다. 만일 사용되고 있다면, 프로세스 플로(1100)은 "B"(도 12의 프로세스 플로(1200)에 관련됨)로 진행되며, 이는 보안 게스트 액세스들에 대해 디폴트 셋팅을 사용한다. 만일 사용되고 있지 않다면, 프로세스 플로(1100)은 블록(1114)으로 진행한다. 블록(1114)에서, 적절한 보안 게스트 도메인이 SG-보안-도메인(SG-secure-domain) 레지스터로 로드 된다(그리고 도 12의 프로세스 플로(1200)에 연결된 "B"로 진행한다).

[0084] [0105] 만일 보안 인터페이스 컨트롤 스토리지에 대한 액세스라면, 프로세스는 플로(1100)은 블록(1120)으로 진행한다(YES 화살표로 도시된 바와 같이). 블록(1120)에서, 액세스는 보안-UV(secure-UV)로 태그 된다(예: UV-보안-도메인(UV-secure-domain) 레지스터를 사용함).

[0085] [0106] 그런 다음 프로세스 플로(1100)은 결정 블록(1130)으로 진행하며, 여기서 보안 인터페이스 컨트롤은 이 것이 UVV 공간에 대한 액세스인지를 결정한다(예: SG-구성(SG-Config) 가변 테이블). 만일 그 것이 UVV 공간에 대한 액세스라면, 프로세스 플로(1100)은 블록(1134)으로 진행한다(YES 화살표로 도시된 바와 같이). 블록(1134)에서, 액세스는 가상으로 태그 된다. 블록(1136)에서, 해당(applicable) 보안 게스트 도메인이 UV-보안-도메인 레지스터에 로드 된다. 블록(1138)에서, DAT 변환 및 액세스 스토리지를 시작하기 위해 준비가 된다. 결정 블록(1130)으로 돌아가면, 만일 UVV 스토리지에 대한 액세스가 아니면, 프로세스 플로(1100)는 블록(1140)을 진행한다(NO 화살표로 도시된 바와 같이). 블록(1140)에서, 액세스는 실제로 태그 된다.

[0086] [0107] 블록(1150)에서, 보안 인터페이스 컨트롤은이 것이 UVS 공간(예: SG 구성 또는 CPU 테이블)에 대한 액세스인지를 결정한다. 만일 UVS 공간에 대한 액세스라면, 프로세스 플로(1100)는 블록(1136)으로 진행한다(YES 화살표로 도시된 바와 같이). 만일 UVS 공간에 대한 액세스가 아니면, 프로세스 플로(1100)은 블록(1170)으로 진행한다(NO 화살표로 도시된 바와 같이). 이 액세스는 UV2 공간(예: 존-보안 테이블)에 대한 액세스 일 수 있다. 블록(1170)에서, 고유한 UV2 보안 도메인이 UV 보안 도메인 레지스터에 로드 된다.

[0087] [0108] 도 12는 본 발명의 하나 또는 그 이상의 실시예들에 따른 프로세스 플로(1200)을 도시한다. 게스트가 디스패치 될 때, SIE 엔트리(Entry) 펌웨어는 게스트가 실행 중임을 하드웨어에 표시할 수 있고(예: 게스트 모드 활성) 게스트가 보안인지를 표시할 수 있다. 만일 게스트가 보안이면, 연관된 보안 게스트 도메인이 하드웨어에(예: SG-보안-도메인 레지스터에) 로드 될 수 있다. 프로그램이 스토리지를 액세스할 때, 하드웨어는 액세스 당시 프로그램의 현재 상태에 기초하여 액세스에 태그 할 수 있다. 도 12는 프로세스 플로(1200)에서 이 프로세스의 예를 예시한다. 블록(1205)에서, 하드웨어는 머신이 현재 게스트 모드에서 실행 중인지를 결정할 수 있고, 만일 게스트 모드에서 실행 중이 아니면, 블록(1210)에서 호스트 액세스인 것으로 액세스에 태그 할 수 있고 블록(1215)에서 비-보안 액세스인 것으로 태그 할 수 있다. 만약 머신이 블록(1205)에서 게스트 모드로 실행 중이라며, 액세스는 블록(1220)에서 게스트 액세스로 태그 될 수 있고 블록(1225)에서 현재 게스트가 보안 게스트인지를 추가로 결정할 수 있다. 만일 게스트가 보안이 아니면, 액세스는 블록(1215)에서 비-안으로 태그 될 수 있다. 만약 게스트가 보안이면, 하드웨어는 블록(1230)에서 게스트를 보안으로 태그 할 수 있으며, 이는 보안 게스트가 디스패치 되었을 때 로드 된 SG-보안-도메인 레지스터와 보안 게스트를 연관시킬 수 있다. 비-보안 게스트와 보안 게스트 모두에 대해, 블록(1235)에서 DAT 상태가 체크될 수 있다. 만일 DAT가 오프(off)이면, 액세스는 블록(1240)에서 실제(real)로 태그 될 수 있다. 만일 DAT가 온(on)이면, 액세스는 블록(1245)에서 가상(virtual)으로 태그 될 수 있다. 일단 상기 액세스가 DAT 오프로 블록(1240)에서 실제로 태그 되거나 또는 상기 액세스가 DAT 온으로 블록(1245)에서 가상으로 태그 되면, 하드웨어는, 블록(1250)에서, DAT변환을 시작하고 스토리지에 액세스 할 준비를 한다. 이에 관해서는 도 13에서 더 설명한다.

[0088] [0109] 도 13은 본 발명의 하나 또는 그 이상의 실시예들에 따른 프로세스 플로(1300)에서 보안 및 비-보안 액세스 모두를 지원하기 위해 하드웨어에 의해 수행되는 변환의 예를 도시한다. 블록(1305)에서, 하드웨어는 액세

스가 게스트 변환으로서 태그 되었는지를 결정할 수 있고, 만일 그렇다면, 블록(1310)에서, 액세스가 가상인지를 결정할 수 있으며, 그 다음에 게스트 DAT가, 블록(1315)에서, 수행될 수 있다. 게스트 DAT 변환 중에, 게스트 DAT 테이블들에 대해 네스트된, 중간 페치들(nested, intermediate fetches)이 있을 수 있다. 상기 테이블 페치들은 게스트 실제로 태그될 수 있고, 만일 원래 변환(the original translation)이 보안으로 태그 되었다면 보안으로 태그 될 수 있다. 상기 테이블 페치들은 또한 프로세스 플로(1300)의 변환 프로세스를 따를 수 있다. 게스트 DAT가, 블록(1315)에서, 게스트 가상으로 태그 된 액세스에 대해 수행되고, 블록(1310)에서, 게스트 실제로 태그 된 모든 액세스에 대해 수행된(가상=아니오) 후, 게스트 프리픽싱(prefixing)과 게스트 메모리 오프셋이, 블록(1320)에서, 적용될 수 있다. 게스트 변환 프로세스가 완료되면, 블록(1325)에서, 최종 주소는 호스트 가상으로 태그 될 수 있고, 만일 원래 게스트 변환이 보안으로 태그 되었다면 보안으로 태그 될 수 있다. 프로세스(1300)는 호스트 가상으로 태그 된 모든 액세스에 대해 계속될 수 있다. 만일 원래 액세스가, 블록(1305)에서, 호스트 액세스(게스트=아님)이고, 블록(1330)에서, 가상이면, 호스트 DAT가, 블록(1335)에서 수행될 수 있다. 호스트 테이블 페치들은, 블록(1335)에서, 비-보안으로 마크 될 수 있다. 블록(1335)에서, 호스트 DAT가 수행된 후, 또는 만일, 블록(1330)에서, 원래 호스트 액세스가 실제(가상=아님)로 태그 되었다면, 호스트 프리픽싱이, 블록(1340)에서 적용될 수 있다. 최종 주소는, 블록(1345)에서, 호스트 절대 주소가 될 수 있다.

[0089] [0110] 도 14는 본 발명의 하나 또는 그 이상의 실시 예들에 따라 프로세스 플로(1400)에서 하드웨어에 의해서 수행될 수 있는 보안 스토리지 보호가 있는 DAT 변환의 예를 도시한다. 도 13의 블록(1345)부터 계속하여, 만일 보안 UV 액세스가 블록(1405)에서 식별되면, 하드웨어는 블록(1410)에서 스토리지가 보안 UV 스토리지로 등록되었는지를 확인(verify)할 수 있으며, 만일 식별되지 않으면, 블록(1415)에서 오류가 제공된다. 보안-UV(secure-UV) 액세스는 UV 스토리지에 액세스할 때 보안 컨트롤 인터페이스에 의해 수행될 수 있다. 만일 스토리지가 블록(1410)에서 보안-UV 스토리지로 등록되었다면, 보호 체크들 (protection checks)은 모든 보안 액세스에 대해 수행될 수 있듯이 계속될 수 있으며, 다만 UV-보안-도메인 레지스터(보안-UV 액세스를 만들기 전에 보안 컨트롤 인터페이스에 의해 셋업(setup)됨)가 처리가 계속되는 블록(1420)에서 도메인 체크를 위해 명시된 보안 도메인으로 사용될 수 있는 경우는 제외한다. 또한, 블록(1425)에서 UV 액세스에 대해 검출된 모든 위반(엔트리 포인트 D)은 블록(1425)에서 보안 게스트 위반에 대해 제공된 것과 같은 블록(1435)에서의 하이퍼바이저에 대해 제공된 예외(보안-UV=No)라기 보다는 블록(1430)에서의 오류로 제공될 수 있다.

[0090] [0111] 블록(1405)에서 보안-UV 액세스로 태그 되지 않은 액세스에 대해서, 하드웨어는 상기 액세스가 보안 게스트 액세스인지를 블록(1440)에서 결정하고, 만일 보안 게스트 액세스가 아니면, 그리고 만일 상기 페이지가 블록(1445)에서 보안으로 마크 되었다면, 예외가 블록(1435)에서 하이퍼바이저로 제공될 수 있다. 그렇지 않고, 만일 상기 액세스가 블록(1440)에서 보안 게스트 액세스가 아니고 상기 페이지가 블록(1445)에서 보안으로 마크 되지 않는다면, 변환은 블록(1450)에서 성공적이 된다.

[0091] [0112] 만일 상기 액세스가 블록(1440)에서의 보안 게스트 액세스 이거나 또는 블록(1410)에서 보안-UV 스토리지로 등록된 스토리지에 대한 보안-UV 액세스라면, 하드웨어는 상기 스토리지가 블록(1420)에서 액세스와 연관된 보안 주체에 등록되는지를 확인하기 위한 체크를 할 수 있다. 만일 이 것이 보안-UV 액세스라면, 상기 명시된 보안-도메인(secure-domain)은 UV-보안 도메인 레지스터(액세스되는 보안-UV 스토리지에 기초하여 보안 컨트롤 인터페이스에 의해 로드 됨)로부터 획득될 수 있고, 보안-게스트(secure-guest) 액세스를 위해, 상기 명시된 보안-도메인은 SG-보안-도메인(SG-secure-domain) 레지스터(보안 주체가 디스패치 될 때 로드 됨)로부터 획득될 수 있다. 만일 액세스 될 스토리지가 블록(1420)에서 상기 명시된 보안-도메인에 등록되어 있지 않다면, 블록(1425)에서의 보안-UV 액세스에 대해서는 오류가 블록(1430)에서 취해지고 블록(1425)에서의 보안-게스트 액세스(보안-UV=No)에 대해서는 예외가 블록(1435)에서 하이퍼바이저에 제공된다.

[0092] [0113] 블록(1420)에서 상기 명시된 보안-도메인에 등록된, 블록(1440) 및 블록(1410)에서의 스토리지에 대한 보안 액세스들에 대해서, 만일 가상 주소 체크가 디스에이블이라면(disabled), 즉 블록(1455)에서 DA-비트(DA-bit)=1이고 상기 액세스가 블록(1460)에서 실제라면, 변환이 블록(1450)에서 완료된다. 그러나, 만일 블록(1455)에서 DA-비트=1이지만 액세스가 블록(1460)에서 가상이면(실제=No), 블록(1425)에서 보안-UV 액세스에 대해 오류가 블록(1430)에서 취해지고 블록(1425)에서 보안-게스트 액세스들(보안-UV=No)에 대해 예외가 블록(1435)에서 하이퍼바이저에 제공된다. 만일 블록(1455)에서 DA-비트=0이고 상기 액세스가 블록(1475)에서 가상 액세스이면, 하드웨어는 상기 액세스의 호스트 가상에서 호스트 절대로의 매핑(the host virtual to host absolute mapping of the access)이 블록(1470)에서 이러한 호스트 절대 주소에 대해 등록된 것과 매치(match)하는지를 결정할 수 있다. 만일 매치한다면, 변환이 블록(1450)에서 성공적으로 완료된다. 만일 상기 매핑이 블록(1470)에서 매치하지 않는다면, 블록(1425)에서 보안-UV 액세스를 위해 오류는 블록(1430)에서 취해지고

블록 1425(보안-UV=No)에서 보안-게스트 액세스들에 대해 예외가 블록(1435)에서 하이퍼바이저에 제공된다. 만일 DA-비트=0이고 상기 액세스가 블록(1475)(가상=No)에서 실제 액세스라면, 블록(1425)에서 보안-UV 액세스들에 대해서는 오류가 블록(1430)에서 취해지고 블록(1425)에서 보안-게스트 액세스들(보안-UV=No)에 대해서는 예외가 블록(1435)에서 하이퍼바이저에 제공된다; 이와 달리(alternately), 변환은 블록(1450)에서 성공적으로 완료될 수도 있다. 블록(1480)에서 I/O 서브시스템에 의한 모든 액세스는 상기 페이지가 블록(1445)에서 보안으로 마크 되었는지를 확인하기 위해 체크될 수 있고, 만일 상기 페이지가 보안이라면, 예외가 블록(1435)에서 하이퍼바이저에 제공될 수 있다; 만일 상기 페이지가 보안으로 마크 되지 않았다면, 변환은 블록(1450)에서 성공적이 된다.

[0093] [0114] 스토리지 등록(registration) 및 매핑에 관한 다양한 체크들이 존 보안 테이블 인터페이스(1485)를 통해 총괄적으로(collectively) 관리될 수 있다. 예를 들어, 블록들(1410, 1420, 1455, 1470 및 1475))은 다양한 액세스들을 관리하기 위해 동일 존과 연관된 존 보안 테이블로 인터페이스 될 수 있다.

[0094] [0115] 이제 도 15-16 을 참조하면, 프로세스 플로(1500 및 1600)은 보안 인터페이스 컨트롤 이외의 모든 주체에 의한 액세스를 금지 하기 위해 태깅 메커니즘(예: 신뢰할 수 없는 엔터티에 의해 시작되고 보안 인터페이스 컨트롤에 의해 마크 됨)에 의해서 스토리지를 보안 인터페이스 컨트롤 메모리로 그리고 보안으로 마크하는 것(marking)과, 상기 보안 인터페이스 컨트롤 메모리를, 보안 게스트들 간의 격리를 위해 하드웨어에 제공된 상기 스토리지 보안 메커니즘에 의해서, 격리하는 것(isolating)에 일반적으로 관련된다.

[0095] [0116] 도 15는 본 발명의 하나 또는 그 이상의 실시 예들에 따른 보안 인터페이스 컨트롤 보안 스토리지 하드웨어 태깅을 위한 프로세스 플로(1500)을 도시한다. 프로세스 플로(1500)은 블록(1510)에서 시작하며, 여기서 신뢰할 수 없는 주체는 보안 인터페이스 컨트롤에 쿼리(query)를 발행한다. 예를 들어, 상기 쿼리(query)는, 쿼리 UVC와 같은, 명령 호출이다.

[0096] [0117] 블록(1520)에서, 보안 인터페이스 컨트롤은 신뢰할 수 없는 주체에 데이터를 리턴 한다. 상기 데이터는, 쿼리 UVC와 같은, 명령 호출로 리턴 될 수 있다. 상기 데이터는 적어도 기부될 스토리지의 양을 포함할 수 있다. 예를 들어, 상기 적어도 기부될 스토리지의 양은 다음 중에서 하나 또는 그 이상을 포함할 수 있다; 베이스 존-특정 스토리지 크기(base zone-specific storage size); 베이스 보안-게스트-도메인-특정 스토리지 크기(base secure-guest-domain-specific storage size); (MB당) 가변 보안-게스트-도메인-특정 스토리지 크기(variable secure-guest-domain-specific storage (per MB)); 및/또는 베이스 보안-게스트-CPU-특정 스토리지 크기(base secure-guest-CPU-specific storage size).

[0097] [0118] 블록(1530)에서, 신뢰할 수 없는 주체는 보안 인터페이스 컨트롤에 대해 초기화(an initialization)를 발행한다(issue). 예를 들어, 상기 초기화는 보안 인터페이스 컨트롤 초기화 UVC(Initialize Secure Interface Control UVC)와 같은, 보안 인터페이스 컨트롤에 스토리지를 기부하는 명령 호출이다. 상기 명령 호출은 신뢰할 수 없는 주체가 베이스 존-특정 스토리지(쿼리 UVC에 의해서 리턴 된 길이)를 유보하게(reserve) 할 수 있다. 상기 명령 호출의 일부는 기본 존별 스토리지 출처 및 길이도 명시할 수 있음에 유의한다.

[0098] [0119] 블록(1540)에서, 보안 인터페이스 컨트롤은 초기화를 구현한다. 예를 들어, 상기 초기화는 기부된 스토리지를 보안으로 적어도 세트 하는 신뢰할 수 없는 주체에 의한 명령 호출(an instruction call)이다. 명령 호출의 예들은 보안 인터페이스 컨트롤 UVC 초기화 또는 UVC 초기화(Initialize Secure Interface Control UVC or Initialize UVC)이 될 수 있다. 또한 UVC 초기화는 보안 인터페이스 컨트롤로 하여금 기부된 스토리지가 현재 비-보안임을 확인하고; 기부된 스토리지를 보안으로 세트 하며; 그리고 베이스 존-특정 스토리지를 UV2에 할당(예: 존-특정 UV 호스트 절대 스토리지)하게 해준다.

[0099] [0120] 블록(1550)에서, 신뢰할 수 없는 주체는 구성 생성(a create configuration)을 발행한다. 예를 들어, 상기 구성 생성은 보안 인터페이스 컨트롤에 스토리지를 기부하는(donate) 명령 호출이다. 이와 관련하여, 이 명령 호출은 보안 게스트 구성 생성 또는 SG 구성 생성(a Create Secure Guest Config or Create SG Config)일 수 있다. 또한, 신뢰할 수 없는 주체는 베이스 보안-게스트-도메인-특정 스토리지(base secure-guest-domain-specific storage)(쿼리 UVC에 의해서 리턴 된 길이)를 유보하고(reserve), 보안-게스트-도메인 스토리지의 크기에 기초한 가변 보안-게스트 도메인-특정 스토리지(variable secure-guest-domain-specific storage based on size of secure-guest-domain storage) (쿼리 UVC에 의해서 리턴 한 MB 당 길이)를 유보하며, 기본 및 가변 보안-게스트-도메인-특정 스토리지 출처 및 길이(base and variable secure-guest-domain-specific storage origin and length)를 명시한다.

[0100] [0121] 프로세스 플로(1500)는 그 다음 도 16의 프로세스 플로(1600)와 연결되는 원 M으로 진행한다.

[0101] [0122] 이제 도 16으로 돌아가면, 보안 인터페이스 컨트롤 보안 스토리지 하드웨어 태깅을 위한 프로세스 플로(1600)는, 도 15의 프로세스 플로의 원 M에서 계속되며, 본 발명의 하나 또는 그 이상의 실시 예들에 따라 도시된다.

[0102] [0123] 블록(1660)에서, 보안 인터페이스 컨트롤은 구성 생성(the create configuration)을 구현한다. 연산에서, 상기 구성 생성은 기부된 스토리지를 보안 인터페이스 컨트롤에 등록하고 이 기부된 스토리지를 보안으로 마크한다. 상기 구성 생성은 보안 게스트 구성 UVC 생성 또는 SG 구성 UVC 생성(a Create Secure Guest Config UVC or Create SG Config UVC)과 같은 명령 호출일 수 있다. SG 구성 UVC 생성을 위해, 보안 인터페이스 컨트롤은 기부된 스토리지가 현재 비-보안임을 확인하고; 기부된 스토리지를 보안으로 세트 하며; 베이스 보안-게스트-도메인-특정 스토리지를 UVS에 할당(예: UV 보안-게스트-도메인-특정 호스트 절대 스토리지)하고, 그리고 가변 보안-게스트-도메인-특정 스토리지를 UVV에 할당(예: UV 보안-게스트-도메인-특정 호스트 가상 스토리지)할 수 있다.

[0103] [0124] 블록(1670)에서, 신뢰할 수 없는 주체는 CPU 생성(a create CPU)을 발행한다. 예를 들어, CPU 생성은 스토리지를 보안 인터페이스 컨트롤에 기부하는 명령 호출이다. 이와 관련하여, 이 명령 호출은 보안 게스트 CPU 생성 또는 SG CPU 생성(a Create Secure Guest CPU or Create SG CPU) 일 수 있다. 예를 들어, 신뢰할 수 없는 주체는 베이스 보안-게스트-CPU-특정 스토리지(base secure-guest-CPU-specific storage)(쿼리 울트라바이저 UVC에 의해서 리턴 된 길이)를 유보하고, 베이스 보안-게스트-CPU-특정 스토리지 출처와 길이를 명시한다.

[0104] [0125] 블록(1680)에서, 보안 인터페이스 컨트롤은 CPU 생성을 구현한다. 예를 들어, CPU 생성은 기부된 스토리지를 보안 인터페이스 컨트롤에 등록하고 그것을 보안으로 마크하는 명령 호출이다. 이와 관련하여, 이 명령 호출은 보안 게스트 CPU UVC 생성 또는 SG CPU UVC 생성(a Create Secure Guest CPU UVC or Create SG CPU UVC)이 될 수 있다. 예를 들어, 보안 인터페이스 컨트롤은 기부된 스토리지가 현재 비-보안임을 확인하고; 기부된 스토리지를 보안으로 세트 하며; 그리고 베이스 보안-게스트-도메인-특정 스토리지를 UVS에 할당(예: UV 보안-게스트-도메인-특정 호스트 절대 스토리지)한다.

[0105] [0126]본 명세서는 클라우드 컴퓨팅에 관해서 상세한 설명들을 포함하지만, 여기서 설명된 기술적 사상들의 구현은 클라우드 컴퓨팅 환경에만 한정되는 것은 아님을 이해해야 한다. 오히려, 본 발명의 실시예들은 지금 알려져 있거나 또는 나중에 개발될 모든 다른 유형의 컴퓨팅 환경과 함께 구현될 수 있다.

[0106] [0127] 클라우드 컴퓨팅은, 최소한의 관리 노력 또는 서비스 제공자와의 상호작용으로 빠르게 제공 및 해제될 수 있는, 구성 가능한(configurable) 컴퓨팅 자원들(예를 들어, 네트워크들, 네트워크 대역폭, 서버들, 처리, 메모리, 스토리지, 애플리케이션들, VM들, 및 서비스들)의 공유 풀에 대한 편리한 주문형(on-demand) 네트워크 액세스를 가능하게 하는 서비스 전달 모델이다. 이 클라우드 모델은 적어도 5가지의 특성(characteristics), 적어도 3가지 서비스 모델(service models), 및 적어도 4가지 배치 모델(deployment models)을 포함할 수 있다.

[0107] [0128] 클라우드 컴퓨팅 특성들은 다음과 같다:

[0108] [0129] 주문형 셀프-서비스(On-demand self-service): 클라우드 소비자는, 서비스 제공자와의 인적 상호작용을 필요로 하지 않고 필요한 만큼 자동적으로, 서버 시간(server time) 및 네트워크 스토리지 같은 컴퓨팅 기능들을 일방적으로 프로비저닝(provisioning)할 수 있다.

[0109] [0130] 광역 네트워크 액세스(Broad network access): 이질적 씬 또는 씩 클라이언트 플랫폼들(heterogeneous thin or thick client platforms)(예를 들어, 모바일폰, 랩탑, 및 PDA)에 의한 사용을 장려하는 표준 메커니즘들을 통해 액세스되는 기능들을 네트워크를 통해서 이용할 수 있다.

[0110] [0131] 자원 풀링(Resource pooling): 제공자의 컴퓨팅 자원들은 멀티-테넌트 모델(a multi-tenant model)을 이용하여, 각기 다른 물리적 및 가상 자원들을 요구(demand)에 따라 동적으로 할당 및 재할당하면서, 다수의 소비자들에게 서비스할 수 있도록 풀에 넣어둔다(pooled). 소비자는 일반적으로 제공된 자원들의 정확한 위치를 제어할 수 없거나 그에 대한 지식이 없지만 더 높은 추상 레벨에서(예를 들어, 국가, 주, 또는 데이터센터) 위치를 명시할 수 있다는 점에서 위치 독립성이 있다.

[0111] [0132] 기민한 탄력성(Rapid elasticity): 역량들(capabilities)이 기민하게 탄력적으로 프로비저닝되어 (어떤 경우엔 자동으로) 신속히 규모를 확장할 수도 있고(scale out) 그리고 탄력적으로 해제되어 신속히 규모를 축소할 수도 있다(scale in). 소비자에게는 프로비저닝할 수 있는 가능성이 종종 무제한인 것으로 보이고 언제든지

얼마든지 구매할 수 있다.

[0112] [0133] 측정 가능한 서비스(Measured service): 클라우드 시스템은 서비스 유형(예를 들어, 스토리지, 처리, 대역폭, 및 활성 사용자 계정)에 적절한 추상화 레벨에서(at some level of abstraction) 계측 기능을 활용하여 자원 사용을 자동으로 제어하고 최적화한다. 자원 사용량은 모니터되고, 제어되고, 그리고 보고될 수 있으며 이로써 이용하는 서비스의 제공자와 사용자 모두에게 투명성을 제공한다.

[0113] [0134] 서비스 모델들(Service Models)은 다음과 같다:

[0114] [0135] 소프트웨어 서비스(Software as a Service)(SaaS): 소비자에게 제공되는 서비스는 클라우드 인프라스트럭처 상에서 실행되는 제공자의 애플리케이션들을 사용하게 해주는 것이다. 애플리케이션들은 웹 브라우저(예를 들어, 웹기반 이메일) 같은 씬(thin) 클라이언트 인터페이스를 통해 여러 클라이언트 장치들에서 액세스 가능하다. 소비자는 네트워크, 서버, 운영체제, 스토리지, 또는 개별 애플리케이션 성능을 포함하는 하부 클라우드 인프라스트럭처를 관리하거나 제어하지 않는다.

[0115] [0136] 플랫폼 서비스(Platform as a Service)(PaaS): 소비자에게 제공되는 서비스는 제공자에 의해 지원되는 프로그래밍 언어들 및 도구들을 이용하여 생성된 소비자-생성 또는 획득 애플리케이션들을 클라우드 인프라스트럭처에 배치하게 해주는 것이다. 소비자는 네트워크, 서버, 운영체제, 또는 스토리지를 포함하는 하부 클라우드 인프라스트럭처를 관리하거나 제어하지 않지만, 배치된 애플리케이션들에 대해서 그리고 가능한 경우 애플리케이션 호스팅 환경 구성들에 대해서 제어할 수 있다.

[0116] [0137] 인프라스트럭처 서비스(Infrastructure as a Service)(IaaS): 소비자에게 제공되는 서비스는 처리, 스토리지, 네트워크, 및 기타 기본 컴퓨팅 자원들을 제공하여 주는 것이며, 여기서 소비자는 임의의 소프트웨어를 배치 및 실행할 수 있고, 이 소프트웨어에는 운영체제와 애플리케이션들이 포함될 수 있다. 소비자는 하부 클라우드 인프라스트럭처를 관리하거나 제어하지 않지만, 운영체제, 스토리지, 배치된 애플리케이션들에 대해서 제어할 수 있고, 가능한 경우 선택된 네트워킹 컴포넌트들(예를 들어, 호스트 방화벽들)에 대해서 제한적으로 제어할 수 있다.

[0117] [0138] 배치 모델들(Deployment Models)은 다음과 같다:

[0118] [0139] 사설 클라우드(Private cloud): 클라우드 인프라스트럭처는 오직 한 조직(an organization)을 위해서 운영되고, 그 조직 또는 제3자에 의해 관리될 수 있으며 옥내(on-premises) 또는 옥외(on-premises)에 위치할 수 있다.

[0119] [0140] 커뮤니티 클라우드(Community cloud): 클라우드 인프라스트럭처는 여러 조직들에 의해 공유되고 관심사(예를 들어, 선교, 보안 요건, 정책, 및 규정 준수 심사)를 공유하는 특정 커뮤니티를 지원하며, 여러 조직들 또는 제3자에 의해 관리될 수 있으며 옥내(on-premises) 또는 옥외(on-premises)에 위치할 수 있다.

[0120] [0141] 공공 클라우드(Public cloud): 클라우드 인프라스트럭처는 일반 대중 또는 대규모 산업 집단에서 이용할 수 있으며 클라우드 서비스를 판매하는 조직이 소유한다.

[0121] [0142] 하이브리드 클라우드(Hybrid cloud): 클라우드 인프라스트럭처는 둘 또는 그 이상의 클라우드들(사설, 커뮤니티, 또는 공공)이 혼합된 구성이며, 이들은 고유한 주체들로 있지만 데이터 및 애플리케이션 이식가능성(portability)을 가능하게 해주는 표준화된 또는 소유권 있는 기술(예를 들어, 클라우드들 사이의 부하 균형을 위한 클라우드 버스팅(cloud bursting))에 의해 서로 결합되어 있다.

[0122] [0143] 클라우드 컴퓨팅 환경은 무국적(statelessness), 낮은 결합(low coupling), 모듈 방식(modularity), 및 의미적 상호운용성(semantic interoperability)에 집중하는 서비스를 지향한다. 클라우드 컴퓨팅의 중심에는 상호 연결된 노드들의 네트워크를 포함하는 인프라스트럭처가 있다.

[0123] [0144] 이제 도 17을 참조하면, 예시적인 클라우드 컴퓨팅 환경(50)이 도시된다. 도시된 바와 같이, 클라우드 컴퓨팅 환경(50)은 하나 또는 그 이상의 클라우드 컴퓨팅 노드들(10)을 포함하며, 이들은 예를 들어 개인 휴대 정보 단말기(PDA) 또는 휴대폰(54A), 데스크탑 컴퓨터(54B), 랩탑 컴퓨터(54C), 및/또는 자동차용 컴퓨터 시스템(54N)과 같은, 클라우드 소비자가 사용하는 로컬 컴퓨팅 장치들과 통신할 수 있다. 노드들(10)은 서로 통신할 수 있다. 이들은 상기에서 기술된 바와 같은 사설, 커뮤니티, 공공, 또는 하이브리드 클라우드들 또는 이들의 조합 등의 하나 또는 그 이상의 네트워크들에서 물리적으로 또는 가상으로 그룹화될 수 있다(도시되지 않음). 이것은 클라우드 소비자가 로컬 컴퓨팅 장치 상에 자원들을 유지할 필요가 없게 클라우드 컴퓨팅 환경(50)이 인프라스트럭처, 플랫폼들 및/또는 소프트웨어를 서비스로서 제공할 수 있게 해준다. 도 17에 도시된 컴퓨팅 장치

들(54A-N)의 유형들은 단지 예시의 목적으로 기술한 것이며 컴퓨팅 노드들(10)과 클라우드 컴퓨팅 환경(50)은 모든 유형의 네트워크 및/또는 네트워크 주소지정가능 연결을 통해서 (예를 들어, 웹 브라우저를 사용하여) 모든 유형의 컴퓨터화 된 장치와 통신할 수 있다는 것을 이해해야 한다.

[0124] [0145] 이제 도 18를 참조하면, 클라우드 컴퓨팅 환경(50)(도 17)에 의해 제공되는 일 세트의 기능별 추상화 계층들이 도시된다. 도 18에 도시된 컴포넌트들, 계층들, 및 기능들은 단지 예시의 목적이며 본 발명의 실시예들은 이들에 한정되지 않는다는 것을 미리 이해해야 한다. 도시된 바와 같이, 다음의 계층들과 그에 대응하는 기능들이 제공된다:

[0125] [0146] 하드웨어 및 소프트웨어 계층(60)은 하드웨어 및 소프트웨어 컴포넌트들을 포함한다. 하드웨어 컴포넌트들의 예들에는: 메인프레임들(61); RISC(Reduced Instruction Set Computer) 아키텍처 기반 서버들(62); 서버들(63); 블레이드 서버들(64); 스토리지 디바이스들(65); 그리고 네트워크 및 네트워킹 컴포넌트들(66)이 포함된다. 일부 실시 예들에서, 소프트웨어 컴포넌트들은 네트워크 애플리케이션 서버 소프트웨어(67) 및 데이터베이스 소프트웨어(68)를 포함한다.

[0126] [0147] 가상화 계층(70)은 추상화 계층을 제공하며 이로부터 다음의 가상 실체들의 예들이 제공될 수 있다: 가상 서버들(71); 가상 스토리지(72); 가상 사설 네트워크를 포함하는, 가상 네트워크들(73); 가상 애플리케이션들 및 운영체제들(74); 및 가상 클라이언트들(75).

[0127] [0148] 한 예에서, 관리 계층(80)은 아래에 기술하는 기능들을 제공한다. 자원 제공(Resource provisioning)(81)은 클라우드 컴퓨팅 환경에서 작업들을 수행하는 데 이용되는 컴퓨팅 자원들 및 기타 자원들의 동적 조달을 제공한다. 계측 및 가격 책정(Metering and Pricing)(82)은 자원들이 클라우드 컴퓨팅 환경에서 이용될 때 비용 추적, 및 이 자원들의 소비에 대한 요금 청구를 제공한다. 한 예에서, 이 자원들은 애플리케이션 소프트웨어 라이센스를 포함할 수 있다. 보안(Security)은 데이터 및 기타 자원들뿐 아니라 클라우드 소비자들과 작업들에 대한 신원 확인을 제공한다. 사용자 포털(User portal)(83)은 소비자들 및 시스템 관리자들에게 클라우드 컴퓨팅 환경에 대한 액세스를 제공한다. 서비스 레벨 관리(Service level management)(84)는 요구되는 서비스 레벨이 충족되도록 클라우드 컴퓨팅 자원 할당 및 관리를 제공한다. 서비스 레벨 협약서(SLA) 기획 및 충족(planning and fulfillment)(85)은 SLA에 부합하는 예상되는 미래 요건에 맞는 클라우드 컴퓨팅 자원들의 사전-배치(pre-arrangement) 및 조달(procurement)을 제공한다.

[0128] [0149] 워크로드 계층(90)은 클라우드 컴퓨팅 환경이 이용될 수 있는 기능들의 예들을 제공한다. 이 계층에서 제공될 수 있는 워크로드들과 기능들의 예들은 다음과 같다: 맵핑 및 네비게이션(91); 소프트웨어 개발 및 라이프사이클 관리(92); 가상 교실 교육 전달(93); 데이터 분석 처리(94); 트랜잭션 처리(95); 및 보안 스토리지 하드웨어 태깅(96).

[0129] [0150] 이제 도 19으로 돌아가면, 시스템(1900)이 본 발명의 하나 또는 그 이상의 실시 예에 따라 도시된다. 시스템(1900)은, 예를 들어, 노드(10)(예, 호스트 노드)를 포함하는데, 노드(10)은, 예를 들어, 네트워크(165)을 통해 하나 또는 그 이상의 클라이언트 디바이스들(20A-20E)와 직접 또는 간접적으로 통신한다. 노드(10)는 클라우드 컴퓨팅 제공 업체의 데이터 센터 또는 호스트 서버가 될 수 있다. 노드(10)는 하이퍼바이저(12)를 실행하고, 하이퍼바이저(12)는 하나 또는 그 이상의 VM(15)(15A-15N)의 배치를 용이하게 한다. 노드(10)는 하드웨어/펌웨어 계층(11)을 더 포함하고, 하드웨어/펌웨어 계층(11)은 VM들(15A-N) 및 하이퍼바이저(12)에 의해 요구되는 기능들에 대한 직접적인 지원을 제공할 뿐만 아니라 VM들(15)에 하나 이상의 서비스들을 제공한다. 현대의 구현들에서, 하드웨어/펌웨어 계층(11)과 하이퍼바이저(12) 사이에, 하드웨어/펌웨어 계층(11)과 VM들(15) 사이에, 하이퍼 바이저(12)와 VM들(15) 사이에, 그리고 하드웨어/펌웨어 계층(11)을 통해 하이퍼 바이저(12)와 VM들(15) 사이에 통신이 제공된다. 본 발명의 하나 또는 그 이상의 실시예들에 따라, 보안 인터페이스 컨트롤이 하드웨어/펌웨어 계층(11)에 제공되고, 하이퍼바이저(12)와 VM(15) 사이의 직접 통신이 제거된다.

[0130] [0151] 예를 들어, 노드(10)는 클라이언트 장치(20A)가 하나 이상의 가상 머신 (15A-15N)을 배치하는 것을 용이하게 할 수 있다. VM들(15A-15N)은 별개의 클라이언트 장치(20A-20E)로부터의 각각의 요청에 응답하여 배치될 수 있다. 예를 들어, VM(15A)은 클라이언트 장치(20A)에 의해 배치될 수 있고, VM(15B)은 클라이언트 장치(20B)에 의해 배치될 수 있으며, VM(15C)은 클라이언트 장치(20C)에 의해 배치될 수 있다. 노드(10)는 또한 클라이언트가 (VM으로서 실행하지 않고) 물리적 서버를 제공하는 것을 용이하게 할 수 있다. 여기에 설명된 예는 VM의 일부로서 노드(10)의 자원 제공을 구현하지만 설명된 기술 솔루션은 물리적 서버의 일부로 자원을 제공하는 데에도 적용될 수 있다.

[0131] [0152] 예를 들어, 클라이언트 디바이스들(20A-20E)은 개인, 기업, 정부 기관, 기업 내의 부서 또는 기타 주체와 같은, 동일 주체에 속할 수 있고, 노드(10)는 상기 주체의 사설 클라우드로서 운영될 수 있다. 이 경우에, 노드(10)는 주체에 속하는 클라이언트 디바이스들(20A-20E)에 의해 배치된 VM들(15A-15N)만을 호스트한다. 다른 예에서, 클라이언트 디바이스들(20A-20E)은 별개의 주체들에 속할 수 있다. 예를 들어, 제1 주체는 클라이언트 디바이스(20A)를 소유할 수 있는 반면, 제2 주체는 클라이언트 디바이스(20B)를 소유할 수 있다. 이 경우 노드(10)는 서로 다른 주체들로부터의 VM들을 호스트하는 공공 클라우드로 운영될 수 있다. 예를 들어, VM들(15A-15N)은 VM(15A)이 VM(15B)에 대한 액세스를 용이하게 하지 않는 슈라우드 방식으로(in a shrouded manner) 배치될 수 있다. 예를 들어, 노드(10)는 IBM z Systems® Processor Resource/Systems Manager(PR/SM) LPAR(Logical Partition) 기능을 사용하여 VM(15A-15N)을 가릴 수 있다(shroud). PR/SM LPAR과 같은, 이들 기능들은 파티션들 간의 격리를 제공하여 노드(10)가 다른 논리 파티션의 동일한 물리적 노드(10) 상의 다른 주체에 대해 2개 또는 그 이상의 VM들(15A-15N)을 배치하는 것을 용이하게 한다.

[0132] [0153] 클라이언트 디바이스들(20A-20E)로부터 하나의 클라이언트 디바이스(20A)는 노드(10)의 하이퍼바이저(12)에 의한 VM의 배치를 요청하는 컴퓨터, 스마트 폰, 태블릿 컴퓨터, 데스크탑 컴퓨터, 랩탑 컴퓨터, 서버 컴퓨터, 또는 임의의 기타 통신 장치이다. 클라이언트 디바이스(20A)는 네트워크(165)를 통해 하이퍼바이저에 의한 수신을 위해 요청을 보낼 수 있다. VM들(15A-15N)으로부터의 하나의 VM(15A)은, 클라이언트 디바이스들(20A-20e) 중 클라이언트 디바이스(20A)로부터의 요청에 응답하여, 하이퍼바이저(12)가 배치하는 VM 이미지이다. 하이퍼바이저(12)는 VM을 생성하고 실행하는 소프트웨어, 펌웨어 또는 하드웨어일 수 있는 VM 모니터(VMM)이다. 하이퍼바이저(12)는 VM(15A)이 노드(10)의 하드웨어컴포넌트들을 사용하여 프로그램을 실행하고 및/또는 데이터를 저장하는 것을 용이하게 한다. 적절한 기능들 및 수정들을 통해 하이퍼바이저(12)는 IBM z Systems®, Oracle's VM Server, Citrix's XenServer, Vmware's ESX, Microsoft Hyper-V 하이퍼바이저 또는 기타 하이퍼바이저일 수 있다. 하이퍼바이저(12)는 노드(10)에서 직접 실행되는 네이티브 하이퍼바이저이거나 다른 하이퍼바이저에서 실행되는 호스트된 하이퍼바이저일 수 있다.

[0133] [0154] 이제 도 20으로 돌아가서 보면, 본 발명의 실시 예들을 구현하기 위한 노드(10)가 본 발명의 하나 또는 그 이상의 실시 예들에 따라 도시되어 있다. 노드(10)는, 본 명세서에 기술된 바와 같이, 다양한 통신 기술들을 이용하는 컴퓨팅 디바이스 및 네트워크들의 임의의 수 및 조합을 포함 및/또는 채용하는 전자적, 컴퓨터 프레임워크 일 수 있다. 노드(10)는 다른 서비스들로 변경하거나 다른 기능들과 무관하게 일부 기능들을 재구성할 수 있는 능력을 가지고 있어서 쉽게 확장 가능하고 모듈화(scalable, extensible, and modular)될 수 있다.

[0134] [0155] 이 실시예에서, 노드(10)는 하나 또는 그 이상의 중앙 처리 유닛들(CPU들)(2001a, 2001b, 2001c 등)를 포함할 수 있는 프로세서(2001)를 갖는다. 프로세서(2001)는 또한, 처리 회로, 마이크로 프로세서, 컴퓨팅 유닛이라 하며, 이는 시스템 버스(2002)를 통해서 시스템 메모리(2003) 및 다양한 다른 컴포넌트들에 결합되어 있다. 시스템 메모리(2003)은 판독 전용 메모리(ROM)(2004) 및 랜덤 액세스 메모리(RAM)(2005)를 포함한다. ROM(2004)은 시스템 버스(2002)에 결합되고, 그리고 기본 입/출력 시스템 (BIOS)을 포함할 수 있으며, 이는 노드(10)의 특정 기본 기능들을 제어한다. RAM은 읽기-쓰기 메모리이고, 프로세서(2001)에 의한 사용을 위해 시스템 버스(2002)에 결합된다.

[0135] [0156] 도 20의 노드(10)는 하드 디스크(2007)를 포함하는데, 이는 프로세서(2001)에 의해서 실행 가능하고 판독 가능한 유형의 스토리지 매체의 예이다. 하드 디스크(2007)은 소프트웨어(2008) 및 데이터(2009)를 저장한다. 소프트웨어(2008)은 프로세서(2001)에 의해서 노드(10)상에서 실행하기 위한 (도 1-19의 프로세스 플로들과 같은, 프로세스들을 수행하기 위한) 명령들로서 저장된다. 데이터(2009)는 소프트웨어(2008)의 연산들을 지원하고 연산들에 의해 사용되기 위해 다양한 데이터 구조들로 조직된 질적 또는 양적 변수들의 값들의 세트를 포함한다.

[0136] [0157] 도20의 노드(10)는 하나 또는 그 이상의 어댑터들(예를 들어, 하드 디스크 컨트롤러들, 네트워크 어댑터들, 그래픽 어댑터들, 등)을 포함하는데, 이들은 프로세서(2001), 시스템 메모리(2003), 하드 디스크(2007), 그리고 노드(10)의 다른 컴포넌트들 (예를 들어, 주변 장치 및 외부 디바이스들)을 상호 연결하고 이들 사이의 통신들을 지원한다. 본 발명의 하나 또는 그 이상의 실시 예들에서, 하나 또는 그 이상의 어댑터들은 하나 또는 그 이상의 I/O버스들 에 연결될 수 있고, 이들은 중간 버스 브리지를 통해 시스템 버스(2002)에 연결 되며, 상기 하나 또는 그 이상의 I/O버스는, 주변장치 컴포넌트 인터커넥트 (Peripheral Component Interconnect: PCI)와 같은 공동 프로토콜들(common protocols)을 이용할 수 있다.

[0137] [0158] 도시 된 바와 같이, 노드(10)는 키보드(2021), 마우스 (2022), 스피커(2023) 및 마이크(2024)를 시스템 버스(2002)에 상호 연결하는 인터페이스 어댑터(2020)를 포함한다. 노드(10)는 시스템 버스(2002)를 디스플레이 (2031)에 상호 연결하는 디스플레이 어댑터(2030)을 포함한다. 디스플레이 어댑터(2030)(및/또는 프로세서2001)는 GUI(2032)의 디스플레이 및 관리와 같은, 그래픽 성능을 제공하는 그래픽 컨트롤러를 포함할 수 있다. 통신 어댑터(2041)는 시스템 버스(2002)를 네트워크(2050)과 상호 연결하는데, 이는 노드(10)를, 서버(2051) 및 데이터베이스(2052)와 같은, 다른 시스템들, 디바이스들, 데이터, 및 소프트웨어와 통신하도록 인에이블 한다. 본 발명의 하나 또는 그 이상의 실시 예들에서, 소프트웨어 (2008) 및 데이터(2009)의 연산들은 서버(2051) 및 데이터베이스(2052)에 의해서 네트워크(2050) 상에서 구현될 수 있다. 예를 들어, 네트워크(2050), 서버(2051) 및 데이터베이스(2052)는 결합하여, 플랫폼 서비스, 소프트웨어 서비스, 및/또는 인프라스트럭처 서비스(예를 들어, 분산 시스템에서 웹 애플리케이션 프로그램으로서)로서, 소프트웨어 (2008) 및 데이터(2009)의 내부 반복들을 제공할 수 있다.

[0138] [0159] 명세서에 기술된 실시 예들은 반드시 컴퓨터 기술에 근거해야 하고, 특히 VM들을 호스트하는 컴퓨터 서버에 근거해야 한다. 또한, 본 발명의 하나 또는 그 이상의 실시 예들은 컴퓨팅 기술 자체, 특히 VM들을 호스트하는 컴퓨터 서버들의 연산에 대한 개선을 촉진하는데, 이는 VM들을 호스트하는 컴퓨팅 서버들이, 하이퍼바이저조차 보안 VM과 연관된 메모리, 레지스터들, 및 기타 데이터를 액세스하는 것이 금지되는, 보안VM들을 호스트하도록 촉진함으로써 그렇게 할 수 있다. 또한, 본 발명의 하나 또는 그이상의 실시 예들은, 보안 VM과 하이퍼바이저의 분리를 촉진하고 따라서 컴퓨터 서버에 의해 호스트되는 VM들의 보안을 유지하는 것을 촉진하기 위해 하드웨어, 펌웨어(예를 들어, 밀리코드), 신뢰할 수 있는 소프트웨어, 또는 이들의 조합을 포함하는 보안 인터페이스 컨트롤(이를 여기서는 "UV"라 한다)을 사용함으로써 VM호스팅 컴퓨터 서버들의 개선을 위한 중요한 단계들을 제공한다. 상기 보안 인터페이스 컨트롤은 보안을 촉진하기 위해서, 여기서 기술한 바와 같이, VM들의 초기화/종료 동안에 VM상태의 보안에 대한 추가의 실질적인 오버헤드가 없는, 가벼운 중간 연산들(provides lightweight intermediate operations)을 제공한다.

[0139] [0160] 본 발명의 실시 예들은 보안 인터페이스 컨트롤 고-레벨 페이지 관리를 구현하는 시스템, 방법 및/또는 컴퓨터 프로그램 제품 (여기서 시스템)을 포함할 수 있다. 각 설명에서, 엘리멘트들을 위한 식별자들은 다른 도면들의 다른 유사한 엘리멘트들을 위해 재사용됨에 유의한다.

[0140] [0161] 본 발명의 다양한 실시 예들이 연관된 도면들을 참조하여 설명된다. 본 발명의 범위를 벗어남이 없이 본 발명의 대안적인 실시예들이 고안될 수 있다. 다양한 연결들 및 위치 관계들(예를 들어, 위, 아래, 인접 등)는 다음 설명 및 도면에서 엘리멘트들 사이에 설명된다. 이러한 연결들 및/또는 위치 관계들은 달리 명시되지 않는 한 직접적이거나 간접적일 수 있으며, 본 발명은 이와 관련하여 제한하려는 의도가 없다. 따라서 주체들의 결합은 직접적 또는 간접적인 결합을 의미할 수 있으며, 주체들 간의 위치 관계는 직접적 또는 간접적인 위치 관계일 수 있다. 또한, 여기에 설명된 다양한 작업들 및 프로세스 단계들은 여기에 자세히 설명되지 않은 추가 단계들 또는 기능들을 갖는 보다 포괄적인 절차 또는 프로세스에 통합될 수 있다.

[0141] [0162] 다음의 정의들 및 약어들은 명세서 및 청구 범위의 해석에 사용될 수 있다. 본 명세서에서 사용된 용어 "포함하다(comprise)", "포함하는 (comprising)", "포함한다(include)", "포함하는(including)", "갖는다(has) ", "갖는(having)", "포함하다(contain)" 또는 "포함하는(containing)", 또는 이들의 다른 변형들은 비-배타적인 포함(a non-exclusive inclusion)을 커버하기 위한 것이다. 예를 들어, 엘리멘트들의 목록을 포함하는 구성, 혼합물, 공정, 방법, 물품 또는 장치는 반드시 그러한 엘리멘트들에만 제한되는 것은 아니며, 명시적으로 나열되지 않거나 또는 그러한 구성, 혼합물, 공정, 방법, 물품 또는 장치에 고유한 다른엘리멘트들도 포함될 수 있다.

[0142] [0163] 또한, 용어 "예시적인"은 "예, 사례, 예시"라는 의미로 사용되었다. 본 명세서에서 "예시적인" 것으로 설명된 임의의 실시예 또는 설계는 반드시 다른 실시예 또는 설계에 비해 선호되거나 유리한 것으로 해석되어서는 안 된다. "적어도 하나" 및 "하나 또는 그 이상"이라는 용어는 1 보다 크거나 같은 임의의 정수, 즉 1, 2, 3, 4 등을 포함 하는 것으로 이해될 수 있다. 용어 "복수"는 2보다 크거나 같은 정수, 즉 2, 3, 4, 5 등을 포함하는 것으로 이해될 수 있다. 용어 "연결"은 간접적인 "연결"과 직접적인 "연결"을 모두 포함할 수 있다.

[0143] [0164] 용어들 "약", "실질적으로", "대략" 및 이들의 변형들은 출원 당시에 이용 가능했던 장비에 기초한 특정 양의 측정과 연관된 오차의 정도를 포함하기 위해 의도된 것이다. 예를 들어, "약"은 주어진 값의 ± 8% 또는 5% 또는 2%의 범위를 포함할 수 있다.

[0144] [0165] 본 발명은 시스템, 방법, 및/또는 통합의 모든 가능한 기술적 세부 레벨에서 컴퓨터 프로그램 제품 일

수 있다. 컴퓨터 프로그램 제품은 프로세서로 하여금 본 발명의 실시 예들을 수행하게 하기 위한 컴퓨터 판독 가능 프로그램 명령을 갖는 컴퓨터 판독가능 저장 매체(또는 매체)를 포함할 수 있다.

[0145] [0166] 상기 컴퓨터 판독 가능 스토리지 매체는 명령 실행 장치에 의해 사용될 명령들을 유지 및 저장할 수 있는 유형의(tangible) 디바이스일 수 있다. 상기 컴퓨터 판독 가능 ㅅ토리지 매체는, 예를 들면, 전자 스토리지 디바이스, 자기 스토리지 디바이스, 광 스토리지 디바이스, 전자기 스토리지 디바이스, 반도체 스토리지 디바이스, 또는 전술한 것들의 모든 적절한 조합일 수 있으며, 그러나 이에 한정되지는 않는다. 컴퓨터 판독 가능 스토리지 매체의 더 구체적인 예들의 비포괄적인 목록에는 다음이 포함될 수 있다: 휴대용 컴퓨터 디스켓, 하드 디스크, 랜덤 액세스 메모리(RAM), 판독-전용 메모리(ROM), 소거 및 프로그램가능 판독-전용 메모리(EPROM 또는 플래시 메모리), 정적 랜덤 액세스 메모리(SRAM), 휴대용 컴팩트 디스크 판독-전용 메모리(CD-ROM), 디지털 다용도 디스크(DVD), 메모리 스틱, 플로피 디스크, 천공-카드들 또는 명령들이 히스토리된 홈에 있는 융기된 구조들 같이 머신적으로 인코드 된 장치, 및 전술한 것들의 모든 적절한 조합. 본 명세서에서 사용될 때, 컴퓨터 판독 가능 스토리지 매체는 무선 전파들이나 다른 자유롭게 전파되는 전자기파들, 도파 관이나 기타 전송 매체(예를 들어, 광섬유 케이블을 통해 전달되는 광 펄스들)를 통해 전파되는 전자기파들, 또는 선(wire)을 통해 전송되는 전기 신호들 같이 그 자체로 일시적인(transitory) 신호들로 해석되지는 않는다.

[0146] [0167] 본 명세서에 기술되는 컴퓨터 판독 가능 명령들은, 예를 들어, 인터넷, 근거리 통신망, 광역 통신망 및/또는 무선 네트워크 등의 통신망(네트워크)을 통해 컴퓨터 판독 가능 스토리지 매체로부터 각각 컴퓨팅/처리 디바이스들로 또는 외부 스토리지 디바이스로부터 외부 컴퓨터로 다운로드 될 수 있다. 상기 통신망은 구리 전송 케이블들, 광 전송 섬유들, 무선 전송, 라우터들, 방화벽들, 스위치들, 게이트웨이 컴퓨터들 및/또는 엣지 서버들을 포함할 수 있다. 각 컴퓨팅/처리 장치 내 네트워크 어댑터 카드 또는 네트워크 인터페이스는 상기 통신망으로부터 컴퓨터 판독 가능 프로그램 명령들을 수신하고 그 컴퓨터 판독 가능 프로그램 명령들을 각각의 컴퓨팅/처리 디바이스 내의 컴퓨터 판독 가능 스토리지 매체에 저장하기 위해 전송한다.

[0147] [0168] 본 발명의 연산들을 실행하기 위한 컴퓨터 판독 가능 프로그램 명령들은 Smalltalk, C++ 또는 그와 유사 언어 등의 객체 지향 프로그래밍 언어와 "C" 프로그래밍 언어 또는 그와 유사한 언어 등의 종래의 절차적 프로그래밍 언어들을 포함하여, 하나 또는 그 이상의 프로그래밍 언어들을 조합하여 작성된(written) 어셈블러 명령들, 명령-세트-아키텍처(ISA) 명령들, 머신 명령들, 머신 종속 명령들, 마이크로코드, 펌웨어 명령들, 상태-셋팅 데이터, 집적회로를 위한 구성 데이터, 또는 소스 코드나 목적 코드일 수 있다. 상기 컴퓨터 판독 가능 프로그램 명령들은 전적으로 사용자의 컴퓨터상에서, 부분적으로 사용자의 컴퓨터상에서, 독립형(stand-alone) 소프트웨어 패키지로서, 부분적으로 사용자의 컴퓨터상에서 그리고 부분적으로 원격 컴퓨터상에서 또는 전적으로 원격 컴퓨터나 서버상에서 실행될 수 있다. 위에서 마지막의 경우에, 원격 컴퓨터는 근거리 통신망(LAN) 또는 광역 통신망(WAN)을 포함한 모든 종류의 네트워크를 통해서 사용자의 컴퓨터에 접속될 수 있고, 또는 이 접속은 (예를 들어, 인터넷 서비스 제공자를 이용한 인터넷을 통해서) 외부 컴퓨터에 이루어질 수도 있다. 일부 실시예들에서, 예를 들어 프로그램 가능 로직 회로, 필드-프로그램 가능 게이트 어레이들(FPGA),또는 프로그램 가능 로직 어레이들(PLA)을 포함한 전자 회로는 본 발명의 실시 예들을 수행하기 위해 전자 회로를 맞춤화하도록 상기 컴퓨터 판독 가능 프로그램 명령들의 상태 정보를 활용하여 상기 컴퓨터 판독 가능 프로그램 명령들을 실행할 수 있다.

[0148] [0169] 본 명세서에서는 본 발명의 실시 예들에 따른 방법들, 장치들(시스템들), 및 컴퓨터 프로그램 제품들의 플로 차트 예시도들 및/또는 블록도들을 참조하여 본 발명의 실시 예들을 기술한다. 플로 차트 예시도들 및/또는 블록도들의 각 블록과 플로 차트 예시도들 및/또는 블록도들 내 블록들의 조합들은 컴퓨터 판독 가능 프로그램 명령들에 의해 구현될 수 있다는 것을 이해할 수 있을 것이다.

[0149] [0170] 이들 컴퓨터 판독 가능 프로그램 명령들은 범용 컴퓨터, 특수목적용 컴퓨터, 또는 기타 프로그램가능 데이터 처리 장치의 프로세서에 제공되어 머신(machine)을 생성하고, 그렇게 하여 그 명령들이 상기 컴퓨터 또는 기타 프로그램가능 데이터 처리 장치의 프로세서를 통해서 실행되어, 상기 플로 차트 및/또는 블록도의 블록 또는 블록들에 명시된 기능들/동작들을 구현하기 위한 수단을 생성할 수 있다. 이들 컴퓨터 판독 가능 프로그램 명령들은 또한 컴퓨터 판독 가능 스토리지 매체에 저장될 수 있으며, 컴퓨터, 프로그램가능 데이터 처리 장치 및/또는 기타 디바이스들에 지시하여 명령들이 저장된 상기 컴퓨터 판독 가능 스토리지 매체가 상기 플로 차트 및/또는 블록도의 블록 또는 블록들에 명시된 기능/동작의 특징들을 구현하는 명령들을 포함하는 제조품(an article of manufacture)을 포함하도록 특정한 방식으로 기능하게 할 수 있다.

[0150] [0171] 상기 컴퓨터 판독 가능 프로그램 명령들은 또한 컴퓨터, 기타 프로그램가능 데이터 처리 장치, 또는 다

른 디바이스에 로드 되어, 상기 컴퓨터, 기타 프로그램가능 장치 또는 다른 디바이스에서 일련의 동작 단계들이 수행되게 하여 컴퓨터 구현 프로세스를 생성하며, 그렇게 하여 상기 컴퓨터, 기타 프로그램가능 장치, 또는 다른 디바이스 상에서 실행되는 명령들이 플로 차트 및/또는 블록도의 블록 또는 블록들에 명시된 기능들/동작들을 구현할 수 있다.

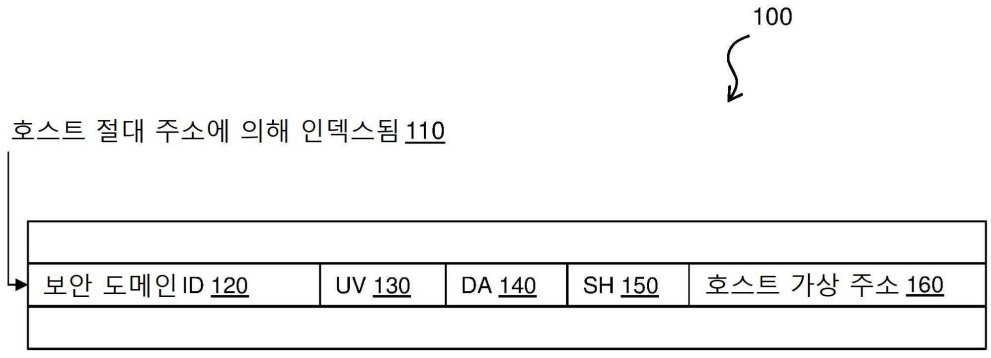
[0151] [0172] 도면들 내 플로 차트 및 블록도들은 본 발명의 여러 실시 예들에 따른 시스템들, 방법들 및 컴퓨터 프로그램 제품들의 가능한 구현들의 아키텍처, 기능(functionality), 및 연산(operation)을 예시한다. 이와 관련하여, 상기 플로 차트 또는 블록도들 내 각 블록은 상기 명시된 논리적 기능(들)을 구현하기 위한 하나 또는 그 이상의 실행 가능한 명령들을 포함한 모듈, 세그먼트 또는 명령들의 일부분을 나타낼 수 있다. 일부 다른 구현들에서, 상기 블록에 언급되는 기능들은 도면들에 언급된 순서와 다르게 일어날 수도 있다. 예를 들면, 연속으로 도시된 두 개의 블록들은 실제로는 사실상 동시에 실행될 수도 있고, 또는 이 두 블록들은 때때로 연관된 기능에 따라서는 역순으로 실행될 수도 있다. 블록도들 및/또는 순서 예시도의 각 블록, 및 블록도들 및/또는 순서 예시도 내 블록들의 조합들은 특수목적용 하드웨어 및 컴퓨터 명령들의 명시된 기능들 또는 동작들, 또는 이들의 조합들을 수행하는 특수목적용 하드웨어-기반 시스템들에 의해 구현될 수 있다는 것에 또한 주목해야 한다.

[0152] [0173] 본 명세서에서 사용된 용어는 단지 특정한 실시 예들을 설명하기위한 것이며 제한을 의도한 것이 아니다. 본 명세서에 사용된 바와 같이, 단수 형태 "a", "an" 및 "the"는 문맥이 명백하게 달리 나타내지 않는 한 복수 형태도 포함하는 것으로 의도된다. 본 명세서에서 사용될 때, "포함하다" 및/또는 "포함하는"이라는 용어는 명시된 특징들, 정수들, 단계들, 연산들, 엘리멘트들 및/또는 컴포넌트들의 존재를 명시하지만, 하나 또는 그 이상의 특징들, 정수들, 단계들, 연산들, 엘리멘트들 및/또는 컴포넌트들 및/또는 이들의 그룹들의 존재 또는 추가를 배제하지 않는다는 것이 추가로 이해될 것이다.

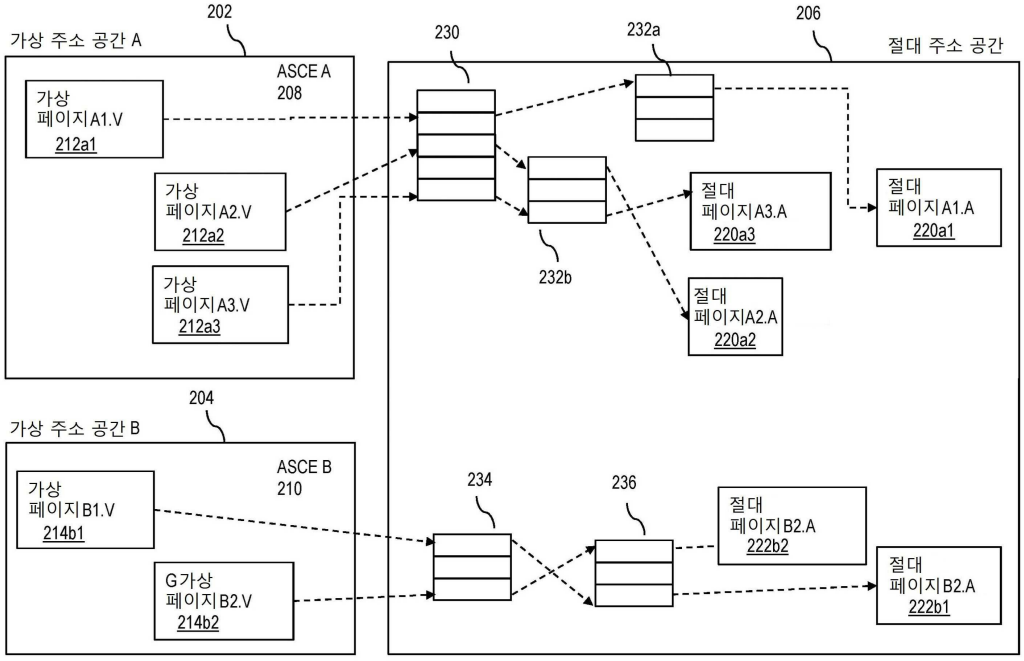
[0153] [0174] 본 발명의 하나 또는 그 이상의 실시 예들에 대한 설명은 예시와 설명의 목적으로 제공되는 것이며, 개시되는 형태로 빠짐없이 총 망라하거나 한정하려는 의도가 있는 것은 아니다. 이 기술 분야에서 통상의 지식을 가진 자들에게 기술된 실시 예들의 범위와 정신을 벗어남이 없이 많은 수정들 및 변형들이 있을 수 있다는 것이 명백하다. 여기서 사용된 용어는 본 발명의 실시 예는 여러 특징들 및 실제 응용을 가장 잘 설명하기 위해 그리고 고려되는 구체적인 용도에 적합하게 여러 수정들을 갖는 다양한 실시 예들을 이 기술 분야에서 통상의 지식을 가진 자들이 이해할 수 있도록 하기 위해, 선택되고 기술되었다.

도면

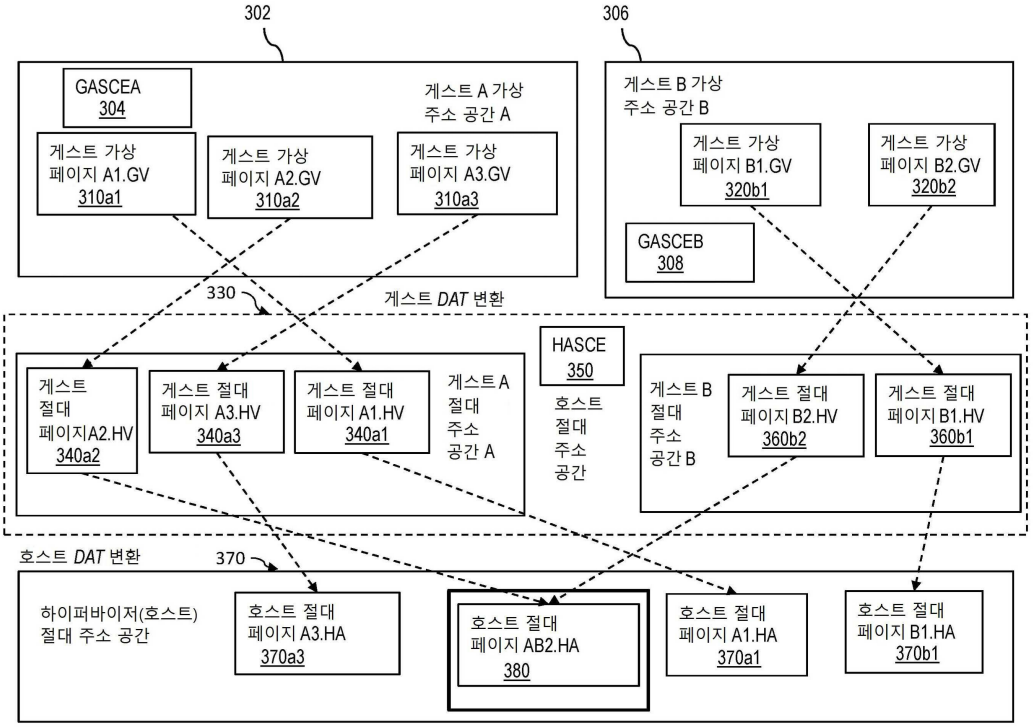
도면1



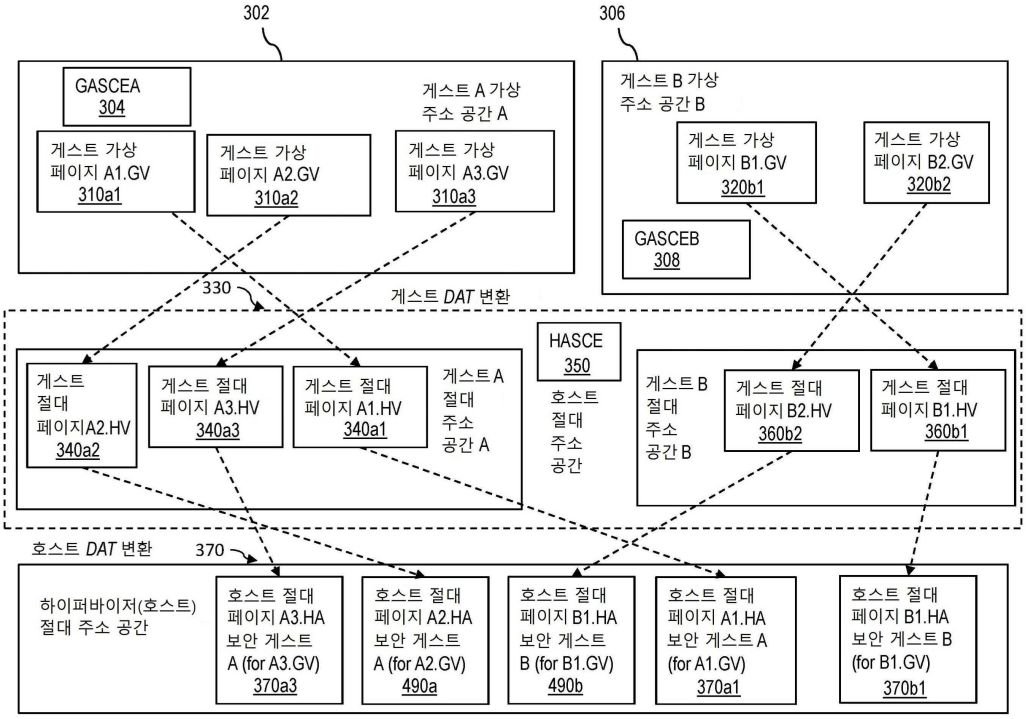
도면2


202
가상 주소 공간 A
ASCE A
208
가상
페이지A1.V
212a1
가상
페이지A2.V
212a2
가상
페이지A3.V
212a3
230
232a
232b
206
절대 주소 공간
절대
페이지A3.A
220a3
절대
페이지A1.A
220a1
절대
페이지A2.A
220a2
204
가상 주소 공간 B
ASCE B
210
가상
페이지B1.V
214b1
G가상
페이지B2.V
214b2
234
236
절대
페이지B2.A
222b2
절대
페이지B2.A
222b1

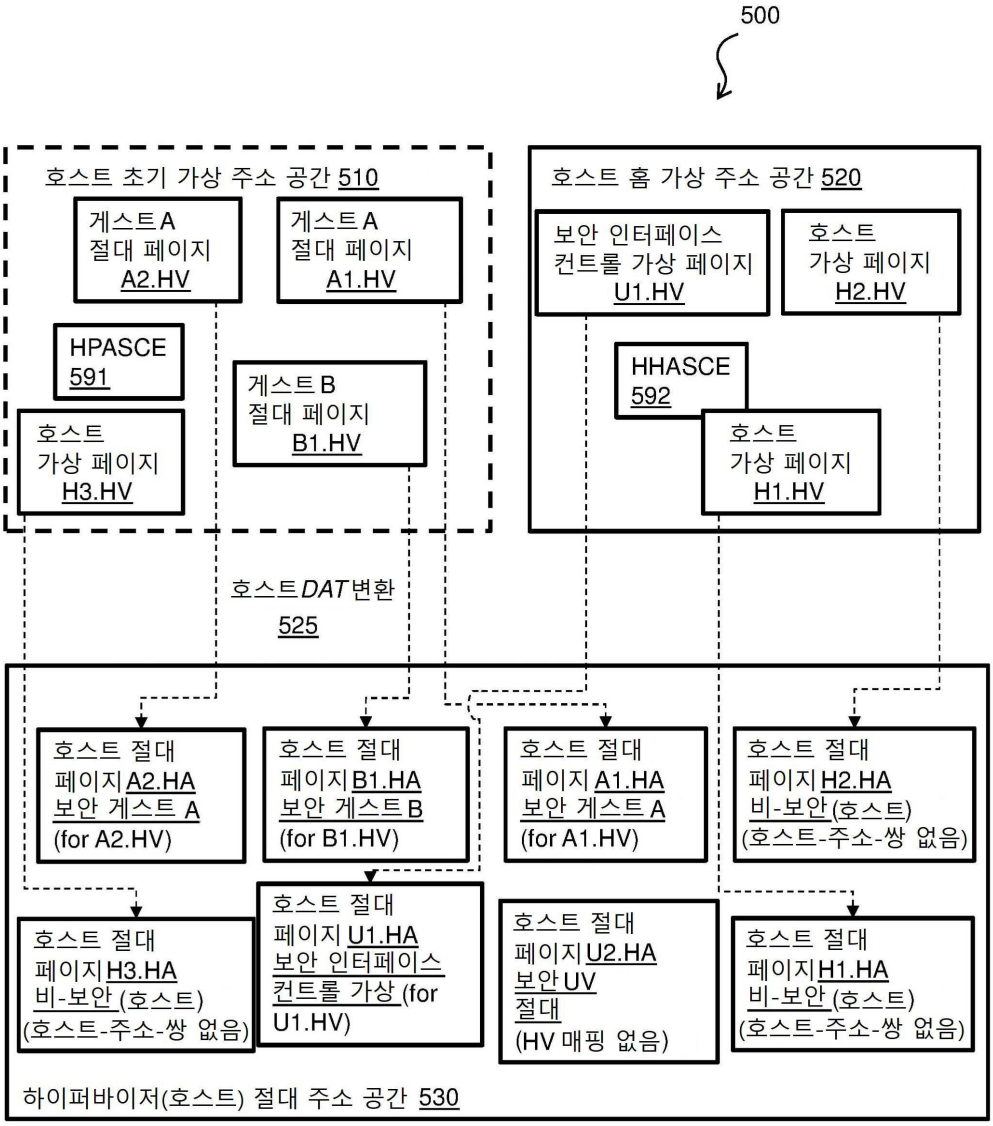
도면3


302
306
GASCEA
304
게스트 A 가상
주소 공간 A
게스트 가상
페이지 A1.GV
310a1
게스트 가상
페이지 A2.GV
310a2
게스트 가상
페이지 A3.GV
310a3
게스트 B 가상
주소 공간 B
게스트 가상
페이지 B1.GV
320b1
게스트 가상
페이지 B2.GV
320b2
GASCEB
308
330
게스트 DAT 변환
HASCE
350
게스트
절대
페이지A2.HV
340a2
게스트 절대
페이지 A3.HV
340a3
게스트 절대
페이지 A1.HV
340a1
게스트 A
절대
주소
공간 A
호스트
절대
주소
공간
게스트 B
절대
주소
공간 B
게스트 절대
페이지 B2.HV
360b2
게스트 절대
페이지 B1.HV
360b1
호스트 DAT 변환
370
하이퍼바이저(호스트)
절대 주소 공간
호스트 절대
페이지 A3.HA
370a3
호스트 절대
페이지 AB2.HA
380
호스트 절대
페이지 A1.HA
370a1
호스트 절대
페이지 B1.HA
370b1

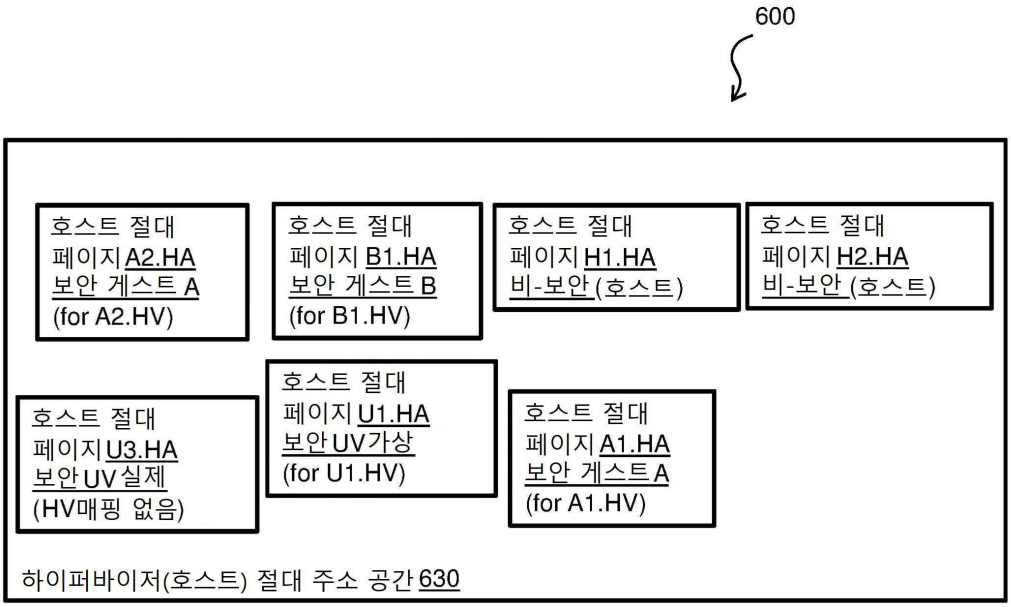
도면4


302
306
GASCEA
304
게스트 가상
페이지 A1.GV
310a1
게스트 가상
페이지 A2.GV
310a2
게스트 가상
페이지 A3.GV
310a3
게스트 A 가상
주소 공간 A
게스트 B 가상
주소 공간 B
게스트 가상
페이지 B1.GV
320b1
게스트 가상
페이지 B2.GV
320b2
GASCEB
308
330
게스트 DAT 변환
HASCE
350
게스트
절대
페이지A2.HV
340a2
게스트 절대
페이지 A3.HV
340a3
게스트 절대
페이지 A1.HV
340a1
게스트 A
절대
주소
공간 A
호스트
절대
주소
공간
게스트 B
절대
주소
공간 B
게스트 절대
페이지 B2.HV
360b2
게스트 절대
페이지 B1.HV
360b1
호스트 DAT 변환
370
하이퍼바이저(호스트)
절대 주소 공간
호스트 절대
페이지 A3.HA
보안 게스트
A (for A3.GV)
370a3
호스트 절대
페이지 A2.HA
보안 게스트
A (for A2.GV)
490a
호스트 절대
페이지 B1.HA
보안 게스트
B (for B1.GV)
490b
호스트 절대
페이지 A1.HA
보안 게스트 A
(for A1.GV)
370a1
호스트 절대
페이지 B1.HA
보안 게스트 B
(for B1.GV)
370b1

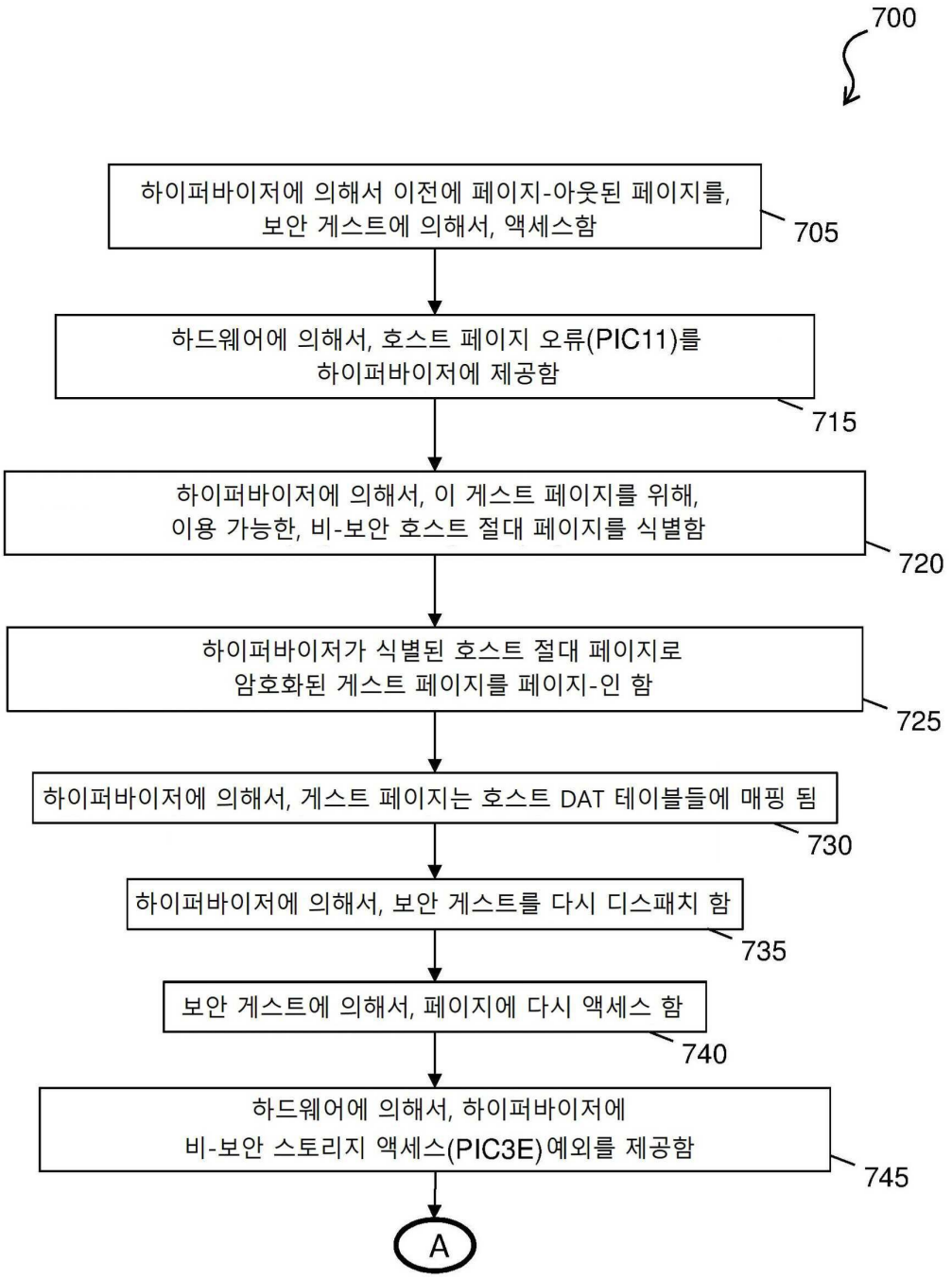
도면5


500
호스트 초기 가상 주소 공간 510
게스트A 절대 페이지 A2.HV
게스트A 절대 페이지 A1.HV
HPASCE 591
게스트B 절대 페이지 B1.HV
호스트 가상 페이지 H3.HV
호스트 홈 가상 주소 공간 520
보안 인터페이스 컨트롤 가상 페이지 U1.HV
호스트 가상 페이지 H2.HV
HHASCE 592
호스트 가상 페이지 H1.HV
호스트DAT변환 525
호스트 절대 페이지A2.HA 보안 게스트 A (for A2.HV)
호스트 절대 페이지B1.HA 보안 게스트B (for B1.HV)
호스트 절대 페이지A1.HA 보안 게스트A (for A1.HV)
호스트 절대 페이지H2.HA 비-보안(호스트) (호스트-주소-쌍 없음)
호스트 절대 페이지H3.HA 비-보안(호스트) (호스트-주소-쌍 없음)
호스트 절대 페이지U1.HA 보안 인터페이스 컨트롤 가상(for U1.HV)
호스트 절대 페이지U2.HA 보안UV 절대 (HV 매핑 없음)
호스트 절대 페이지H1.HA 비-보안(호스트) (호스트-주소-쌍 없음)
하이퍼바이저(호스트) 절대 주소 공간 530

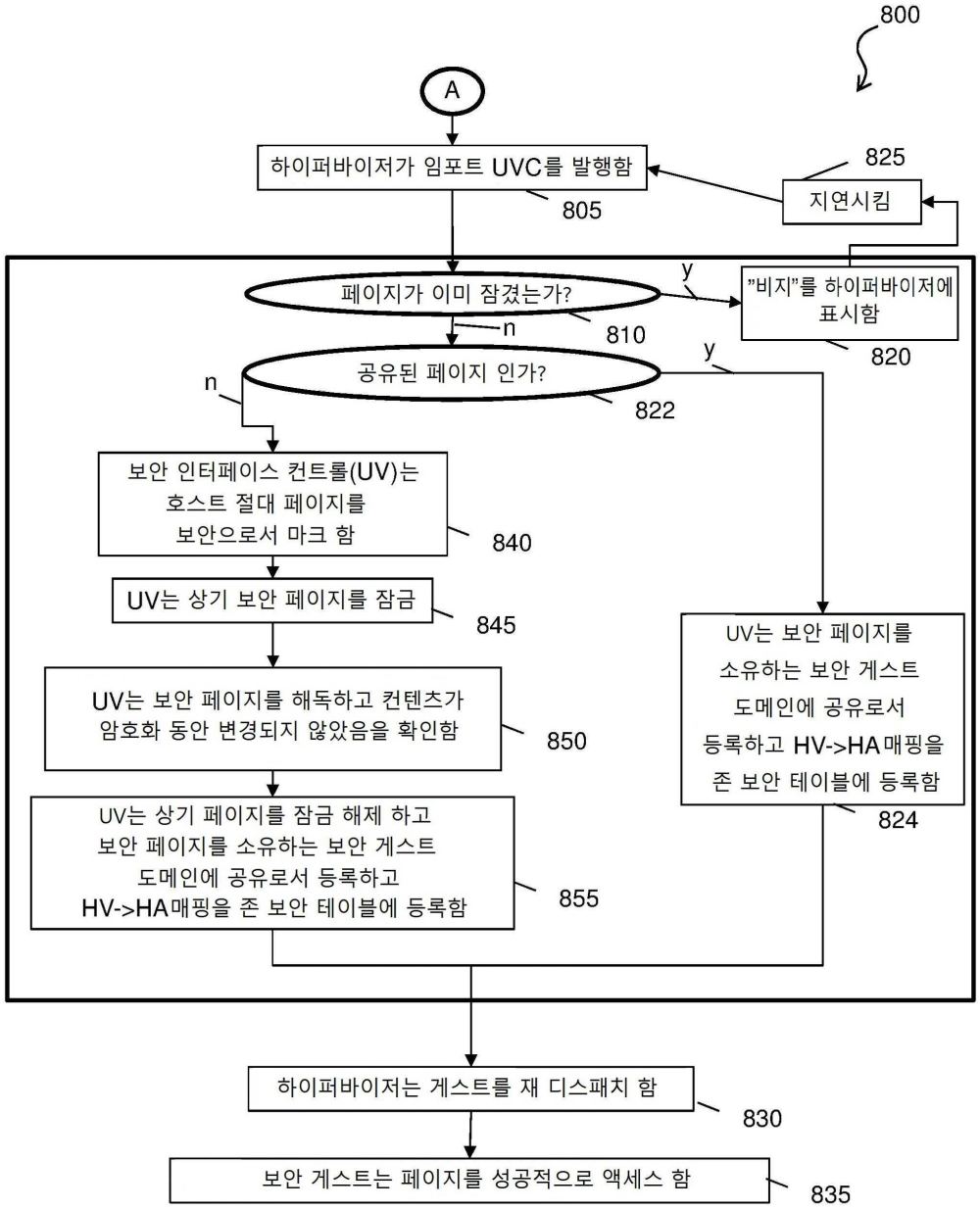
도면6


600
호스트 절대 페이지 A2.HA 보안 게스트 A (for A2.HV)
호스트 절대 페이지 B1.HA 보안 게스트 B (for B1.HV)
호스트 절대 페이지 H1.HA 비-보안 (호스트)
호스트 절대 페이지 H2.HA 비-보안 (호스트)
호스트 절대 페이지 U3.HA 보안 UV 실제 (HV매핑 없음)
호스트 절대 페이지 U1.HA 보안 UV가상 (for U1.HV)
호스트 절대 페이지 A1.HA 보안 게스트 A (for A1.HV)
하이퍼바이저(호스트) 절대 주소 공간 630

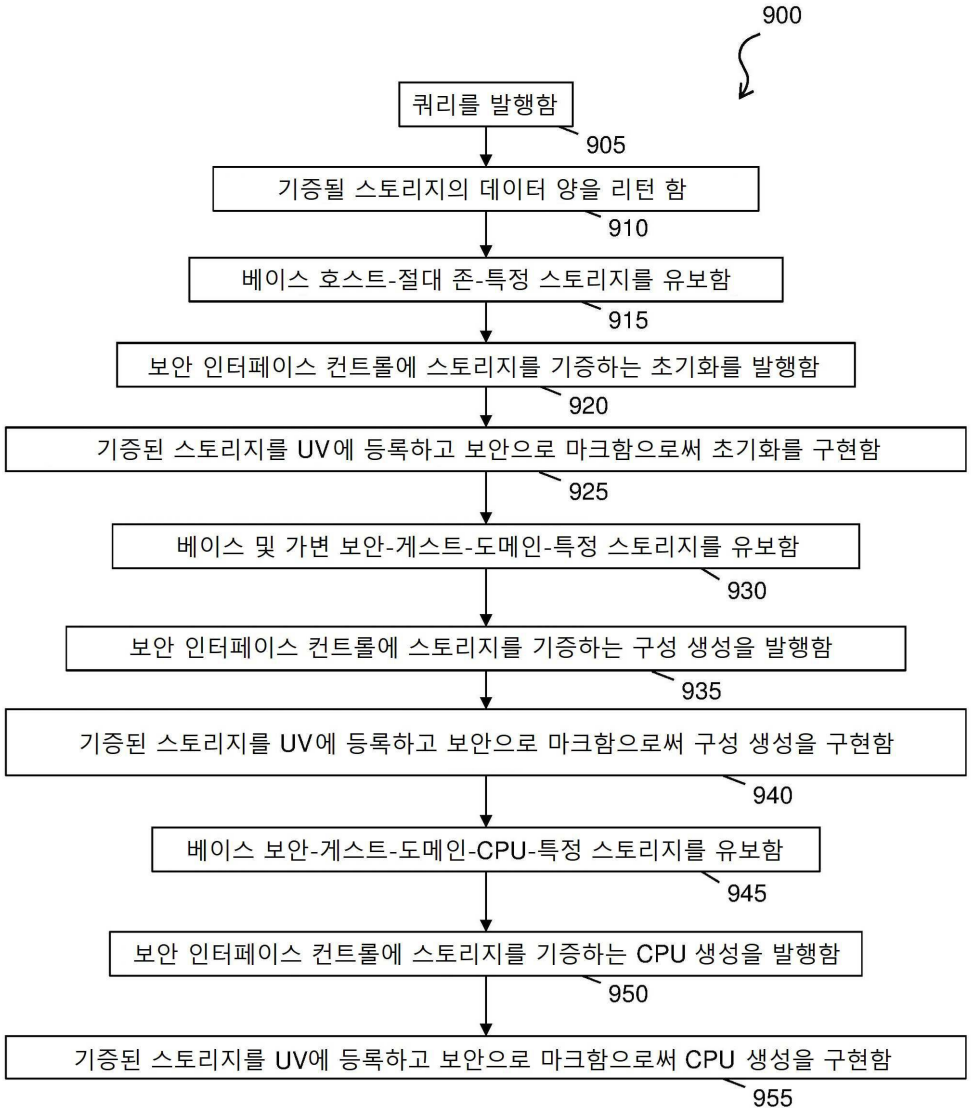
도면7


700
하이퍼바이저에 의해서 이전에 페이지-아웃된 페이지를, 보안 게스트에 의해서, 액세스함
705
하드웨어에 의해서, 호스트 페이지 오류(PIC11)를 하이퍼바이저에 제공함
715
하이퍼바이저에 의해서, 이 게스트 페이지를 위해, 이용 가능한, 비-보안 호스트 절대 페이지를 식별함
720
하이퍼바이저가 식별된 호스트 절대 페이지로 암호화된 게스트 페이지를 페이지-인 함
725
하이퍼바이저에 의해서, 게스트 페이지는 호스트 DAT 테이블들에 매핑 됨
730
하이퍼바이저에 의해서, 보안 게스트를 다시 디스패치 함
735
보안 게스트에 의해서, 페이지에 다시 액세스 함
740
하드웨어에 의해서, 하이퍼바이저에 비-보안 스토리지 액세스(PIC3E)예외를 제공함
745
A

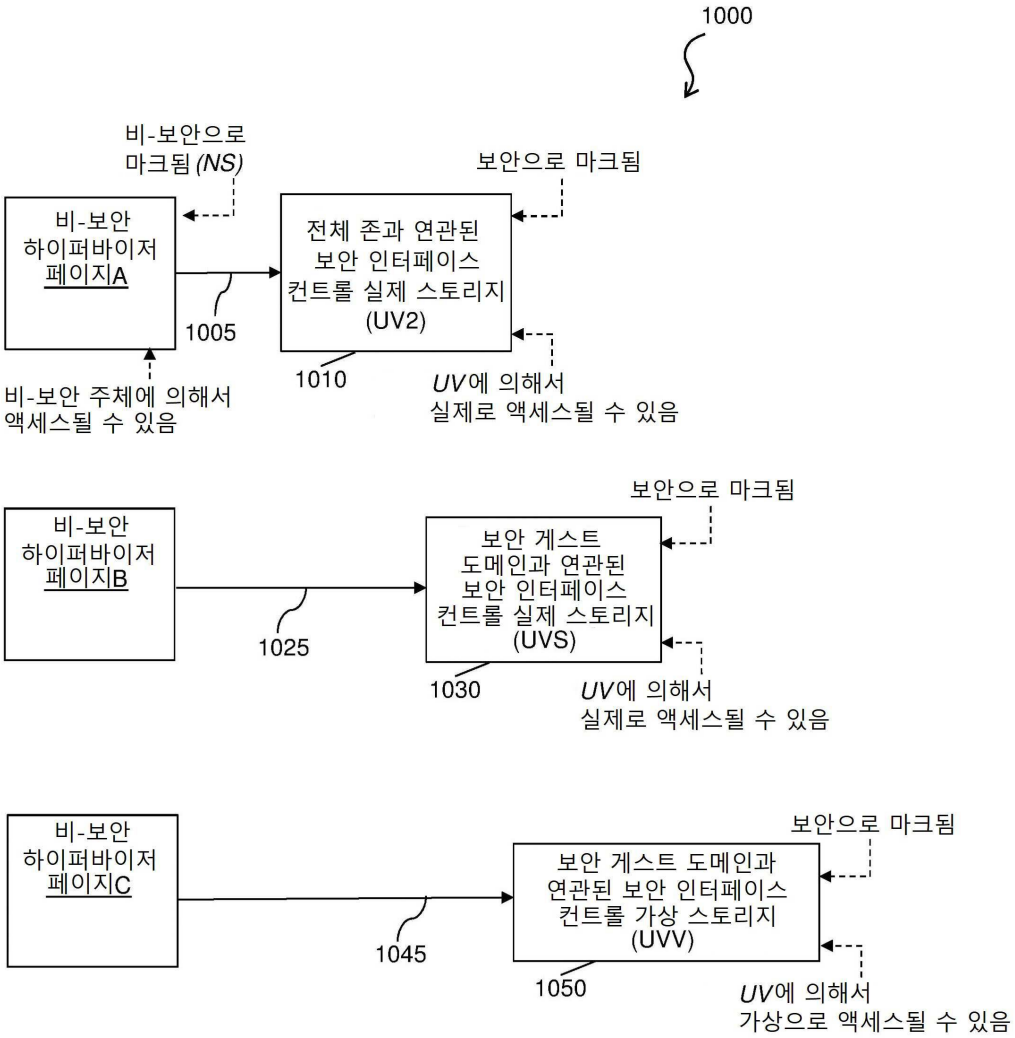
도면8


800
A
하이퍼바이저가 임포트 UVC를 발행함
805
825
지연시킴
페이지가 이미 잠겼는가?
y
n
810
"비지"를 하이퍼바이저에 표시함
820
공유된 페이지 인가?
y
n
822
보안 인터페이스 컨트롤(UV)는 호스트 절대 페이지를 보안으로서 마크 함
840
UV는 상기 보안 페이지를 잠금
845
UV는 보안 페이지를 해독하고 컨텐츠가 암호화 동안 변경되지 않았음을 확인함
850
UV는 보안 페이지를 소유하는 보안 게스트 도메인에 공유로서 등록하고 HV->HA매핑을 존 보안 테이블에 등록함
824
UV는 상기 페이지를 잠금 해제 하고 보안 페이지를 소유하는 보안 게스트 도메인에 공유로서 등록하고 HV->HA매핑을 존 보안 테이블에 등록함
855
하이퍼바이저는 게스트를 재 디스패치 함
830
보안 게스트는 페이지를 성공적으로 액세스 함
835

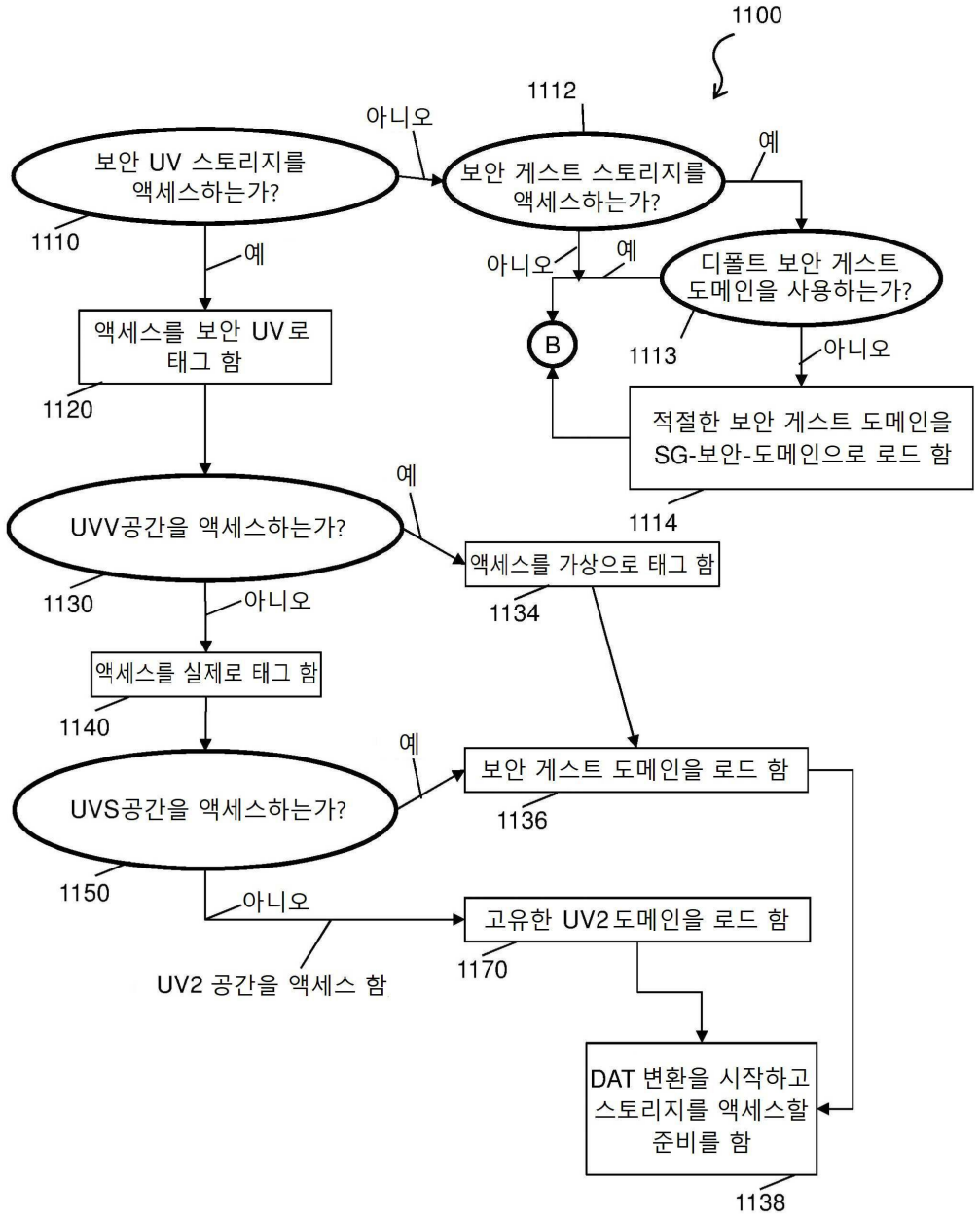
도면9


900
쿼리를 발행함
905
기증될 스토리지의 데이터 양을 리턴 함
910
베이스 호스트-절대 존-특정 스토리지를 유보함
915
보안 인터페이스 컨트롤에 스토리지를 기증하는 초기화를 발행함
920
기증된 스토리지를 UV에 등록하고 보안으로 마크함으로써 초기화를 구현함
925
베이스 및 가변 보안-게스트-도메인-특정 스토리지를 유보함
930
보안 인터페이스 컨트롤에 스토리지를 기증하는 구성 생성을 발행함
935
기증된 스토리지를 UV에 등록하고 보안으로 마크함으로써 구성 생성을 구현함
940
베이스 보안-게스트-도메인-CPU-특정 스토리지를 유보함
945
보안 인터페이스 컨트롤에 스토리지를 기증하는 CPU 생성을 발행함
950
기증된 스토리지를 UV에 등록하고 보안으로 마크함으로써 CPU 생성을 구현함
955

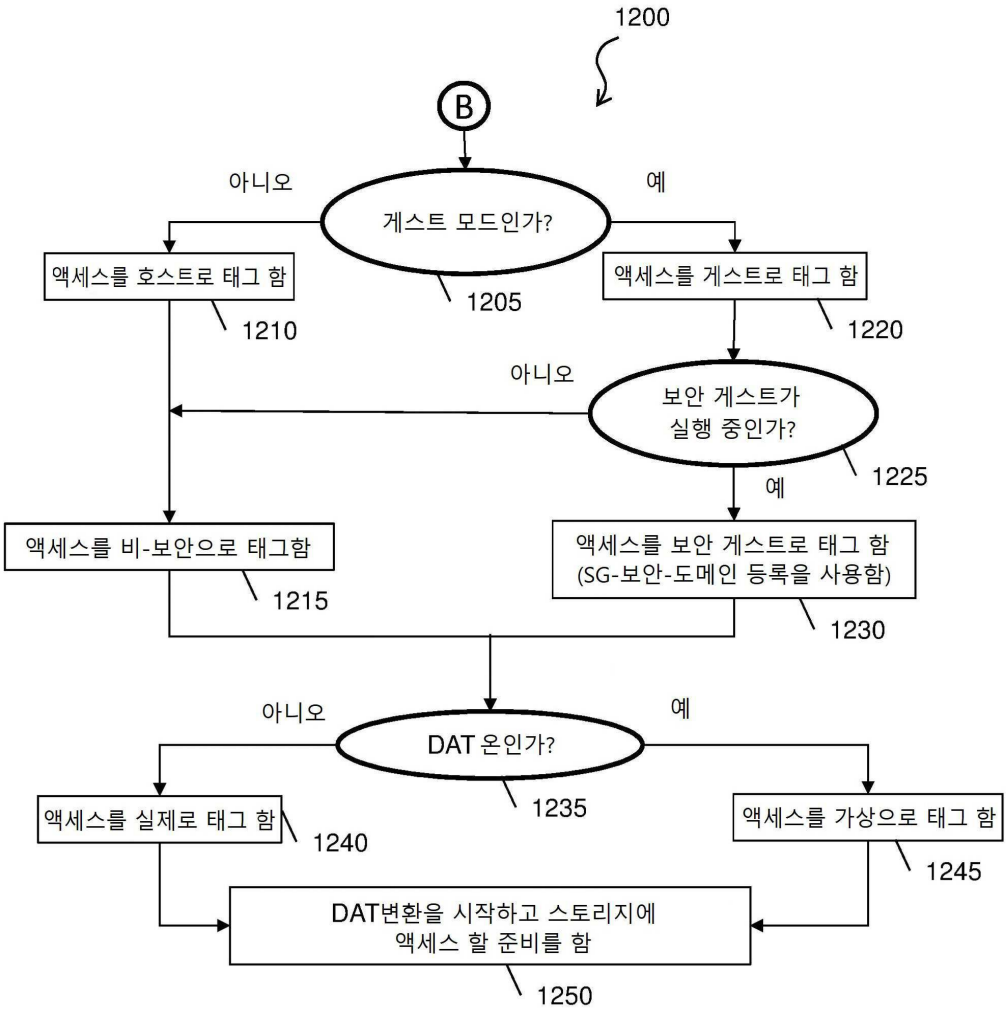
도면10


1000
비-보안으로
마크됨 (NS)
보안으로 마크됨
비-보안
하이퍼바이저
페이지A
전체 존과 연관된
보안 인터페이스
컨트롤 실제 스토리지
(UV2)
1005
1010
비-보안 주체에 의해서
액세스될 수 있음
UV에 의해서
실제로 액세스될 수 있음
보안으로 마크됨
비-보안
하이퍼바이저
페이지B
보안 게스트
도메인과 연관된
보안 인터페이스
컨트롤 실제 스토리지
(UVS)
1025
1030
UV에 의해서
실제로 액세스될 수 있음
보안으로 마크됨
비-보안
하이퍼바이저
페이지C
보안 게스트 도메인과
연관된 보안 인터페이스
컨트롤 가상 스토리지
(UVV)
1045
1050
UV에 의해서
가상으로 액세스될 수 있음

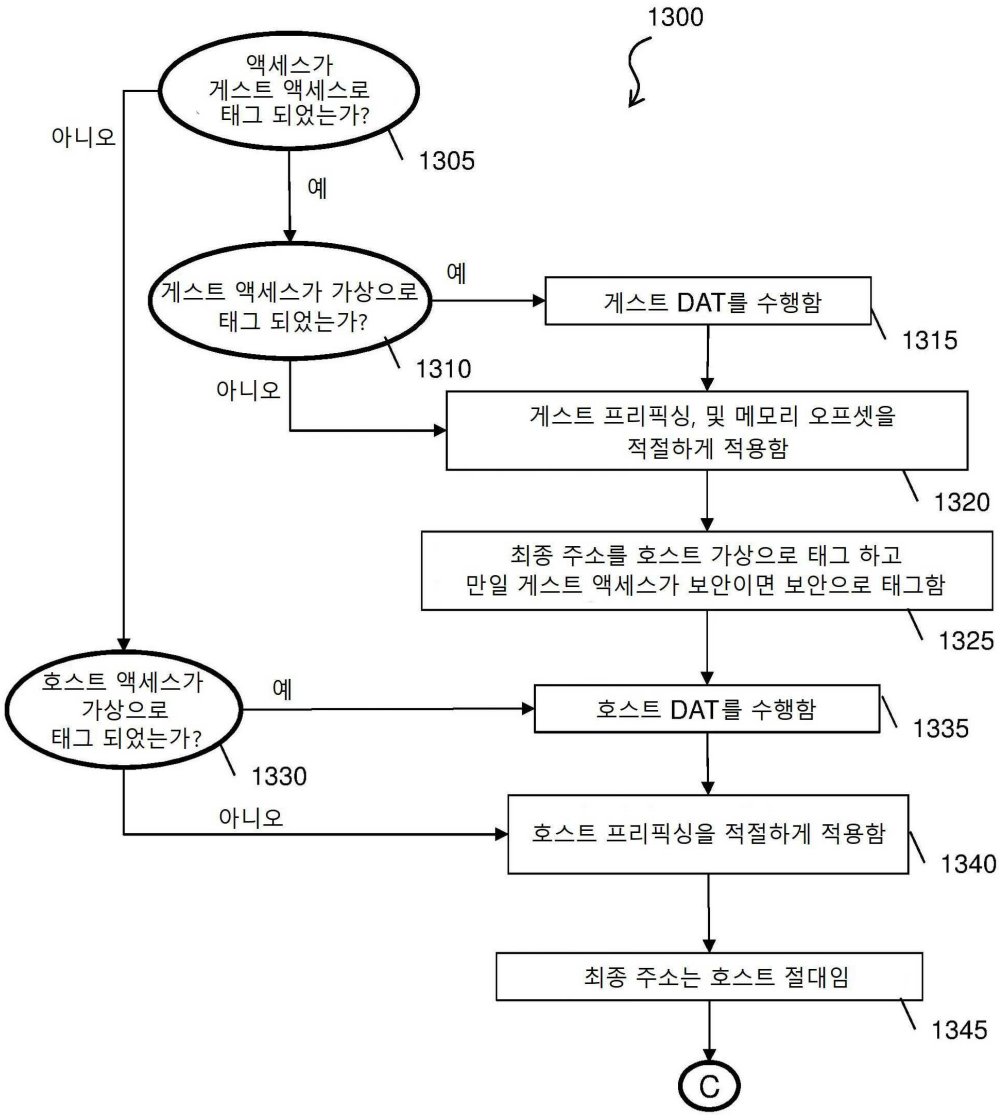
도면11


1100
1112
아니오
보안 UV 스토리지를 액세스하는가?
보안 게스트 스토리지를 액세스하는가?
예
1110
예
아니오
예
디폴트 보안 게스트 도메인을 사용하는가?
액세스를 보안 UV로 태그 함
B
1113
아니오
1120
적절한 보안 게스트 도메인을 SG-보안-도메인으로 로드 함
1114
예
UVV공간을 액세스하는가?
액세스를 가상으로 태그 함
1130
아니오
1134
액세스를 실제로 태그 함
1140
예
보안 게스트 도메인을 로드 함
UVS공간을 액세스하는가?
1136
1150
아니오
고유한 UV2도메인을 로드 함
UV2 공간을 액세스 함
1170
DAT 변환을 시작하고 스토리지를 액세스할 준비를 함
1138

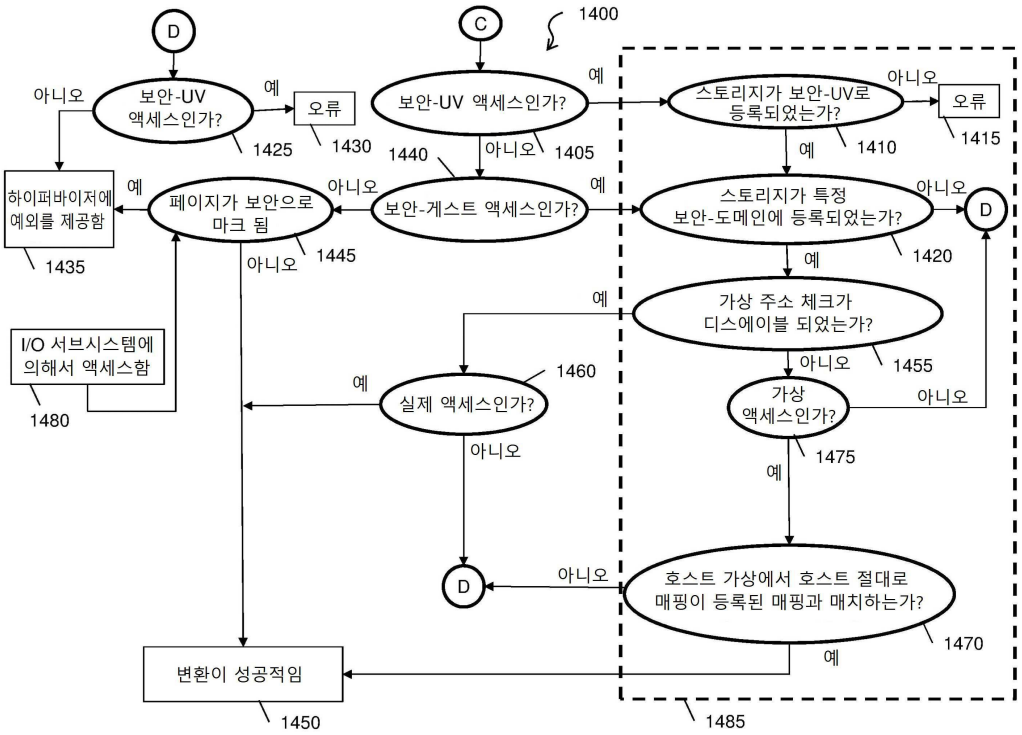
도면12


1200
B
아니오
게스트 모드인가?
예
1205
액세스를 호스트로 태그 함
1210
액세스를 게스트로 태그 함
1220
아니오
보안 게스트가
실행 중인가?
1225
예
액세스를 비-보안으로 태그함
1215
액세스를 보안 게스트로 태그 함
(SG-보안-도메인 등록을 사용함)
1230
아니오
DAT 온인가?
예
1235
액세스를 실제로 태그 함
1240
액세스를 가상으로 태그 함
1245
DAT변환을 시작하고 스토리지에
액세스 할 준비를 함
1250

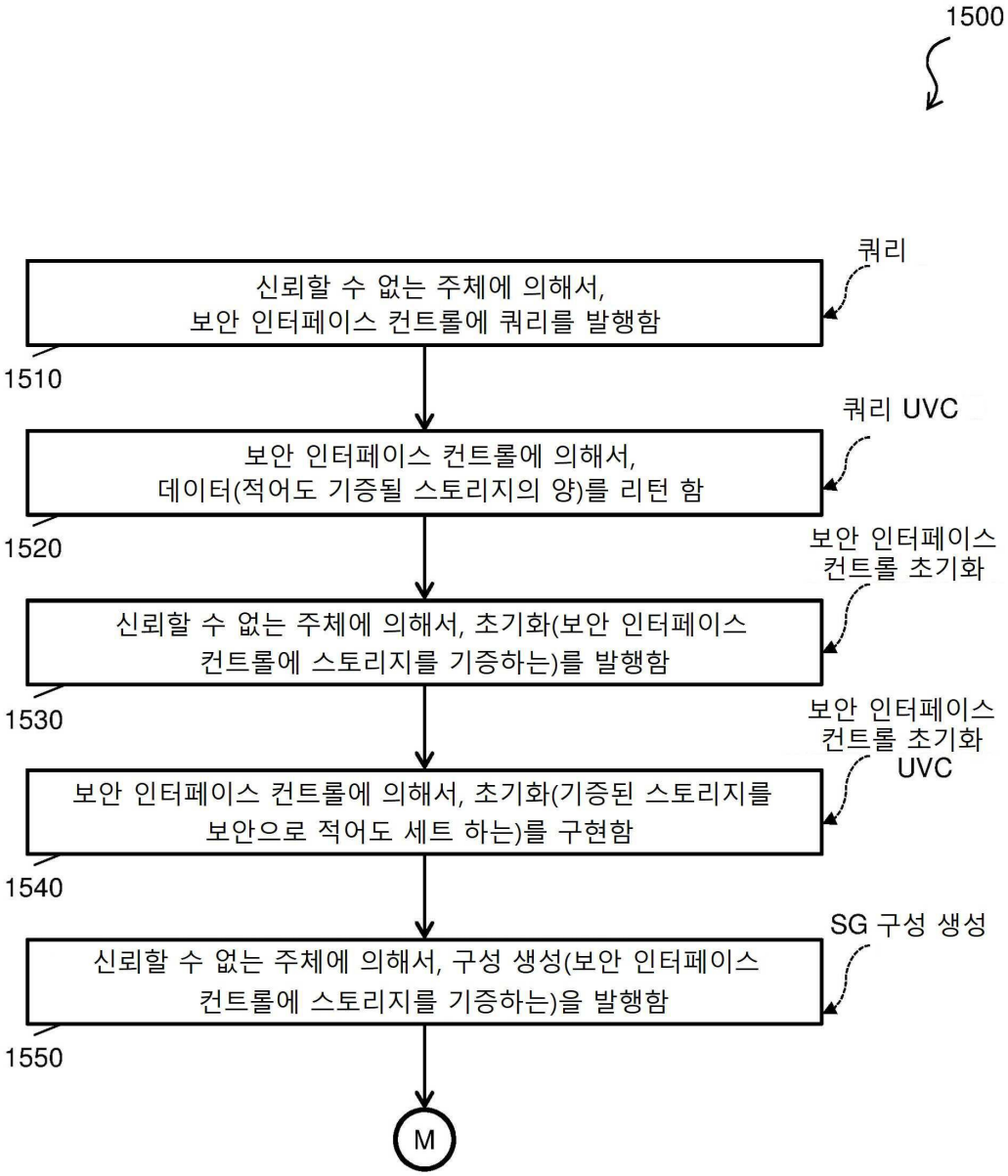
도면13


1300
액세스가 게스트 액세스로 태그 되었는가?
1305
아니오
예
게스트 액세스가 가상으로 태그 되었는가?
1310
예
게스트 DAT를 수행함
1315
아니오
게스트 프리픽싱, 및 메모리 오프셋을 적절하게 적용함
1320
최종 주소를 호스트 가상으로 태그 하고 만일 게스트 액세스가 보안이면 보안으로 태그함
1325
호스트 액세스가 가상으로 태그 되었는가?
1330
예
호스트 DAT를 수행함
1335
아니오
호스트 프리픽싱을 적절하게 적용함
1340
최종 주소는 호스트 절대임
1345
C

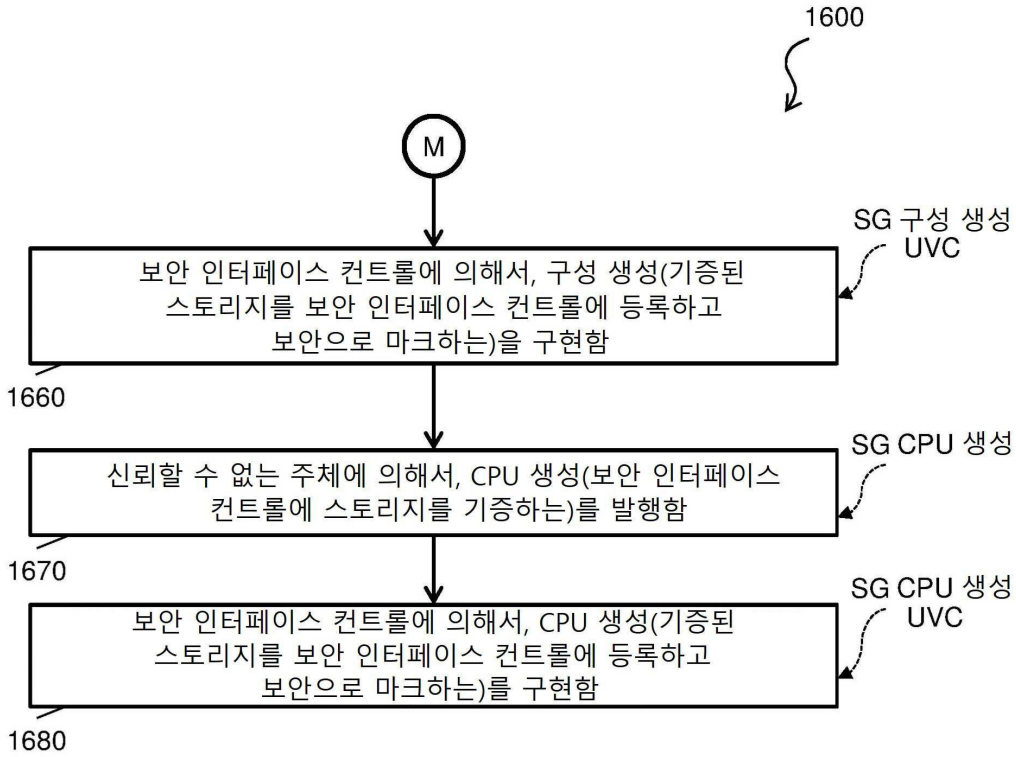
도면14


1400
C
D
보안-UV 액세스인가?
1425
오류
1430
보안-UV 액세스인가?
1405
스토리지가 보안-UV로 등록되었는가?
1410
오류
1415
하이퍼바이저에 예외를 제공함
1435
페이지가 보안으로 마크 됨
1445
보안-게스트 액세스인가?
1440
스토리지가 특정 보안-도메인에 등록되었는가?
1420
D
가상 주소 체크가 디스에이블 되었는가?
1455
I/O 서브시스템에 의해서 액세스함
1480
실제 액세스인가?
1460
가상 액세스인가?
1475
D
호스트 가상에서 호스트 절대로 매핑이 등록된 매핑과 매치하는가?
1470
변환이 성공적임
1450
1485
예
아니오

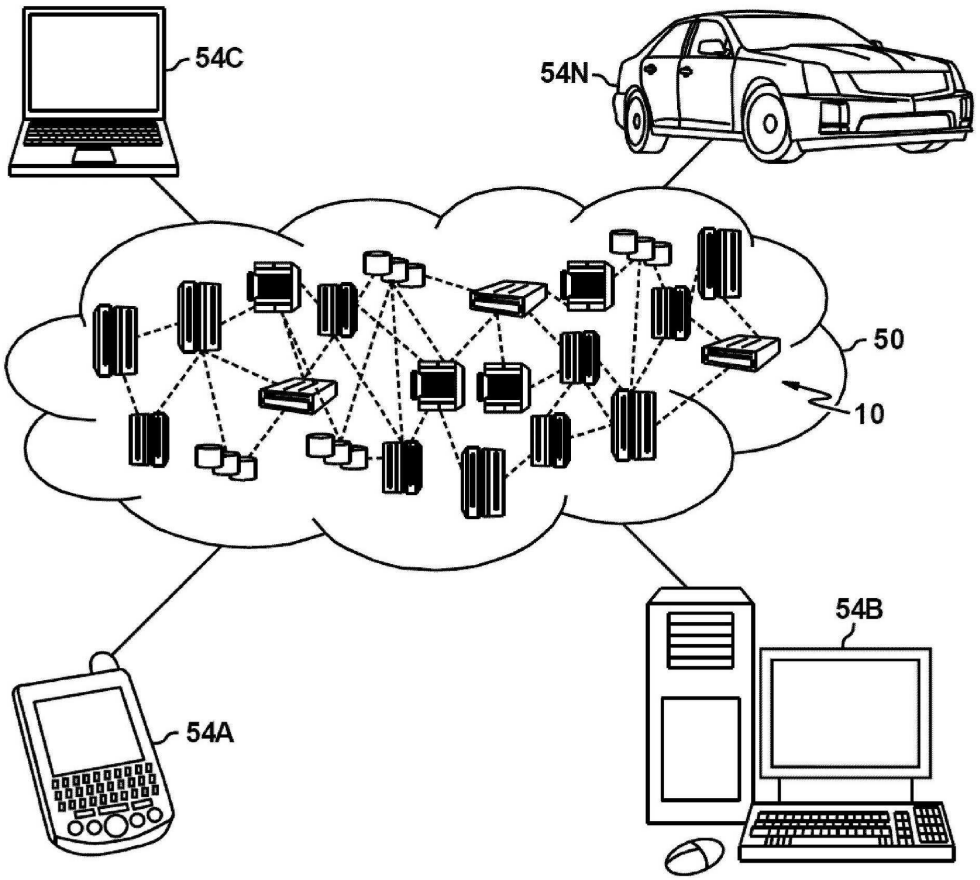
도면15


1500
신뢰할 수 없는 주체에 의해서,
보안 인터페이스 컨트롤에 쿼리를 발행함
쿼리
1510
보안 인터페이스 컨트롤에 의해서,
데이터(적어도 기증될 스토리지의 양)를 리턴 함
쿼리 UVC
1520
신뢰할 수 없는 주체에 의해서, 초기화(보안 인터페이스
컨트롤에 스토리지를 기증하는)를 발행함
보안 인터페이스
컨트롤 초기화
1530
보안 인터페이스 컨트롤에 의해서, 초기화(기증된 스토리지를
보안으로 적어도 세트 하는)를 구현함
보안 인터페이스
컨트롤 초기화
UVC
1540
신뢰할 수 없는 주체에 의해서, 구성 생성(보안 인터페이스
컨트롤에 스토리지를 기증하는)을 발행함
SG 구성 생성
1550
M

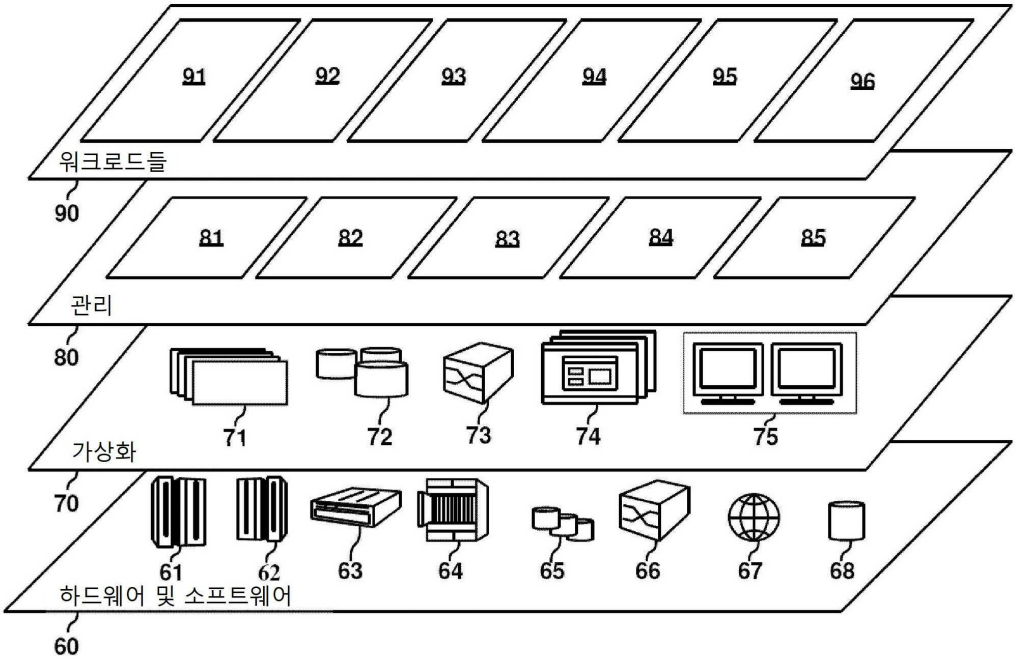
도면16


1600
M
보안 인터페이스 컨트롤에 의해서, 구성 생성(기증된 스토리지를 보안 인터페이스 컨트롤에 등록하고 보안으로 마크하는)을 구현함
SG 구성 생성 UVC
1660
신뢰할 수 없는 주체에 의해서, CPU 생성(보안 인터페이스 컨트롤에 스토리지를 기증하는)를 발행함
SG CPU 생성
1670
보안 인터페이스 컨트롤에 의해서, CPU 생성(기증된 스토리지를 보안 인터페이스 컨트롤에 등록하고 보안으로 마크하는)를 구현함
SG CPU 생성 UVC
1680

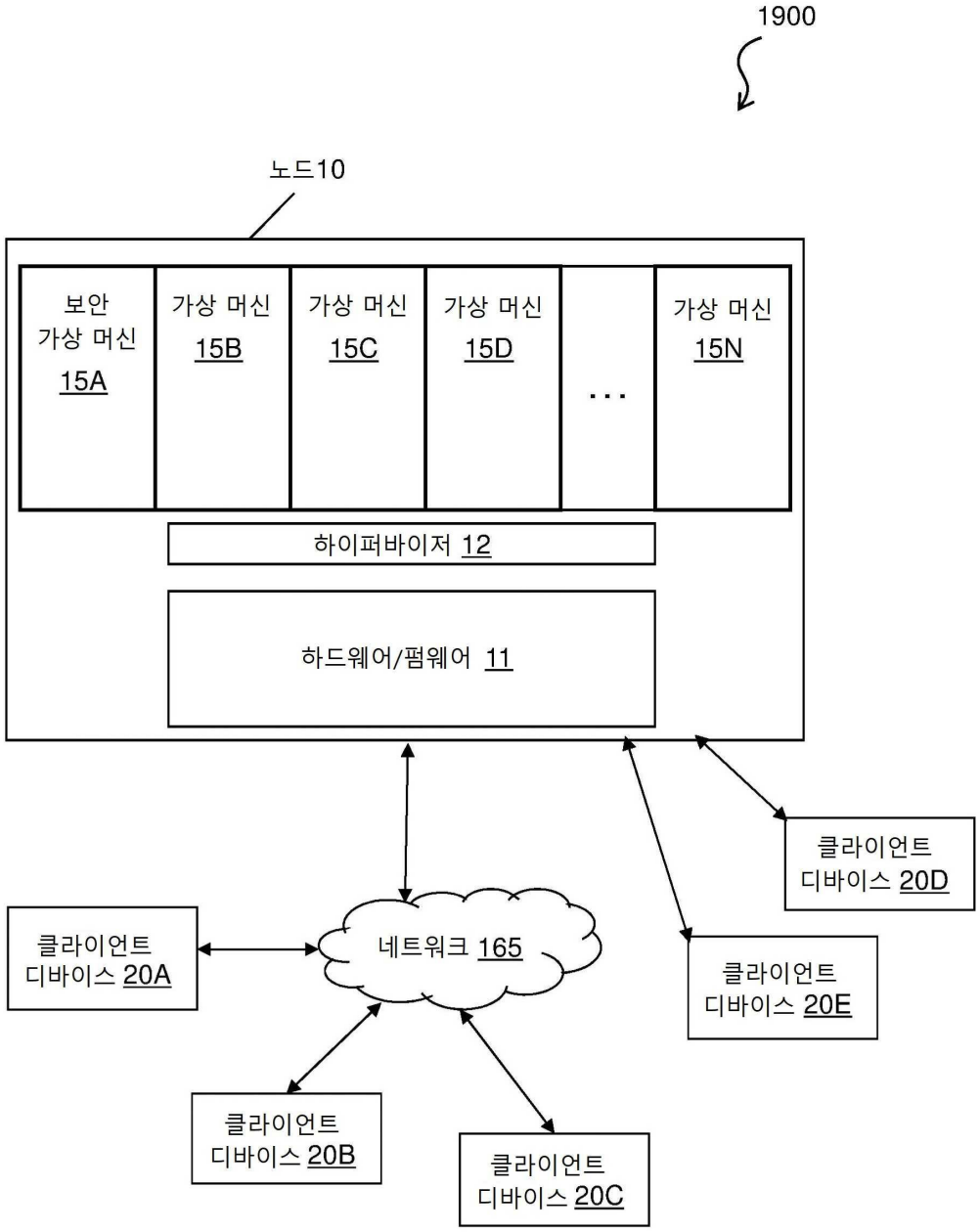
도면17


54C
54N
50
10
54B
54A

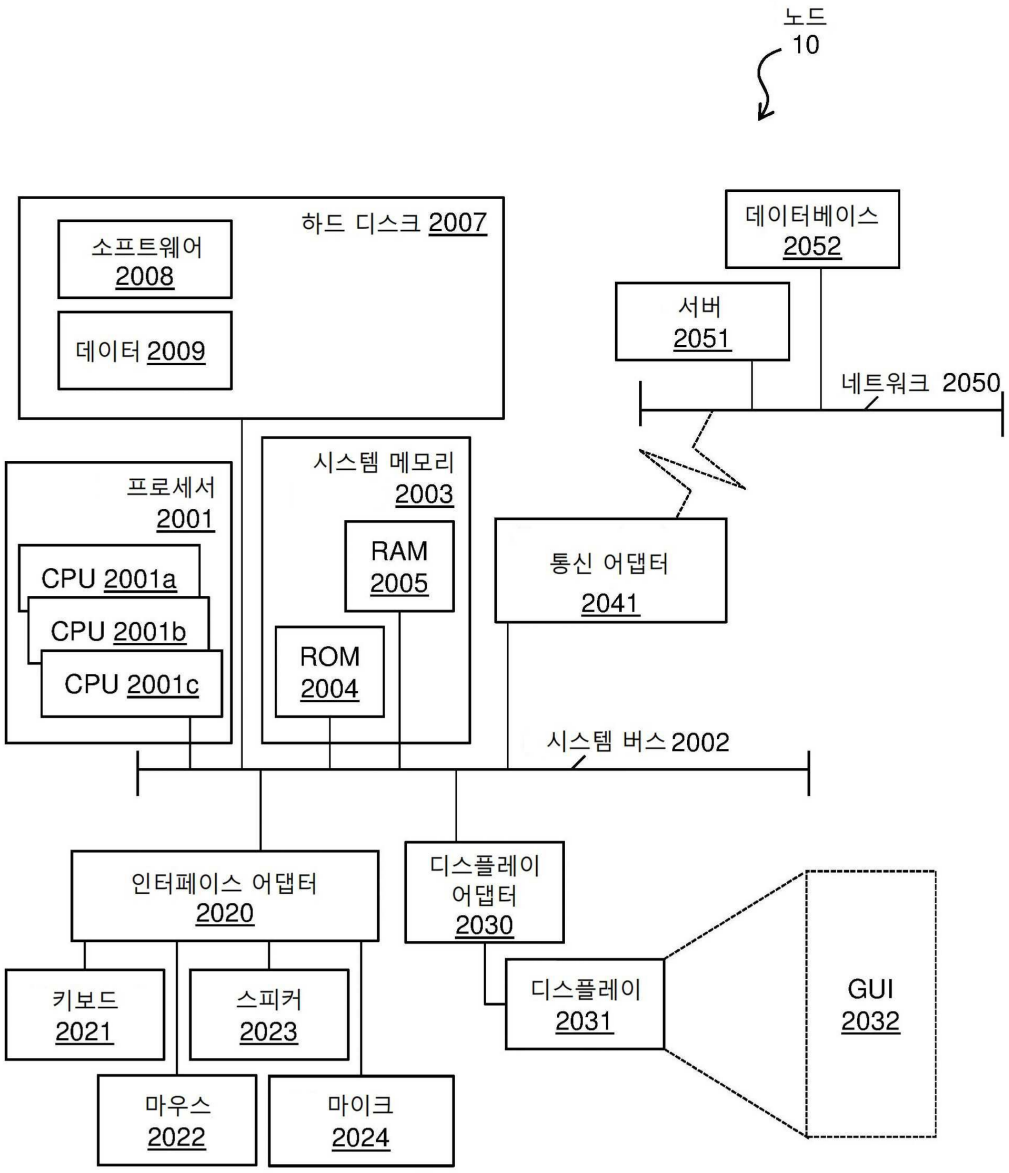
도면18


91
92
93
94
95
96
워크로드들
90
81
82
83
84
85
관리
80
71
72
73
74
75
가상화
70
61
62
63
64
65
66
67
68
하드웨어 및 소프트웨어
60

도면19


1900
노드10
보안
가상 머신
15A
가상 머신
15B
가상 머신
15C
가상 머신
15D
...
가상 머신
15N
하이퍼바이저 12
하드웨어/펌웨어 11
클라이언트
디바이스 20D
클라이언트
디바이스 20A
네트워크 165
클라이언트
디바이스 20E
클라이언트
디바이스 20B
클라이언트
디바이스 20C

도면20


노드
10
하드 디스크 2007
소프트웨어
2008
데이터 2009
데이터베이스
2052
서버
2051
네트워크 2050
프로세서
2001
CPU 2001a
CPU 2001b
CPU 2001c
시스템 메모리
2003
RAM
2005
ROM
2004
통신 어댑터
2041
시스템 버스 2002
인터페이스 어댑터
2020
디스플레이
어댑터
2030
키보드
2021
스피커
2023
디스플레이
2031
GUI
2032
마우스
2022
마이크
2024