(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2016년10월06일
(11) 등록번호 10-1663338
(24) 등록일자 2016년09월29일

(51) 국제특허분류(Int. Cl.)
G06Q 50/10 (2012.01) *G06F 21/57* (2013.01)
(21) 출원번호 10-2011-7019717
(22) 출원일자(국제) 2010년01월21일
심사청구일자 2014년12월23일
(85) 번역문제출일자 2011년08월25일
(65) 공개번호 10-2011-0126122
(43) 공개일자 2011년11월22일
(86) 국제출원번호 PCT/US2010/021563
(87) 국제공개번호 WO 2010/098910
국제공개일자 2010년09월02일
(30) 우선권주장
12/394,430 2009년02월27일 미국(US)
(56) 선행기술조사문헌
US20040078572 A1*
US20050033980 A1*
JP2006323814 A
JP2008546122 A
*는 심사관에 의하여 인용된 문헌

(73) 특허권자
마이크로소프트 테크놀로지 라이센싱, 엘엘씨
미국 워싱턴주 (우편번호 : 98052) 레드몬드 원 마이크로소프트 웨이
(72) 발명자
레이 케네스 디
미국 워싱턴주 98052-6399 레드몬드 원 마이크로소프트 웨이
알코브 제임스 엠
미국 워싱턴주 98052-6399 레드몬드 원 마이크로소프트 웨이
(뒷면에 계속)
(74) 대리인
김태홍

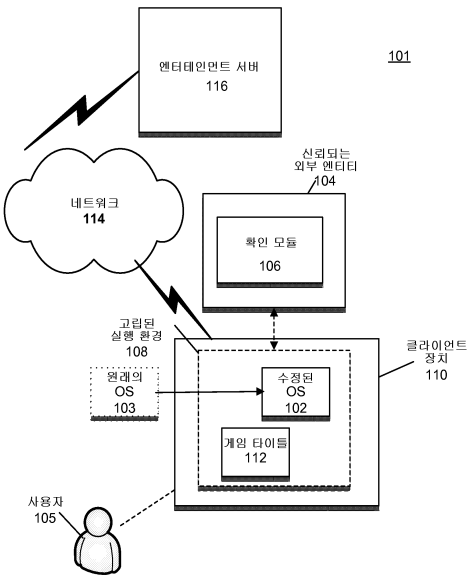
전체 청구항 수 : 총 21 항 심사관 : 박재용

(54) 발명의 명칭 **치팅 방지 방법, 치팅 방지 시스템 및 컴퓨터 판독가능 매체**

(57) 요 약

엔티 치팅 시스템은 수정된 운영 체제와 같은 수정된 환경과, 수정된 환경이 특정 장치에서 실행되고 있음을 확인하기 위한 신뢰성이 있는 외부 엔티티의 결합을 포함할 수 있다. 수정된 환경은, 수정된 환경에 의해 대체되는 원래의 환경에 비해 제한된 환경을 생성하기 위해 특정한 방식으로 수정될 수 있다. 수정된 환경으로의 수정은, 예를 들어, 치팅 또는 바람직하지 않은 사용자 행동을 허용하기 위한 하드웨어 및/또는 소프트웨어에 대한 변경을 검출 및/또는 방지하기 위한 원래 환경에 대한 변경을 포함할 수 있다.

대 표 도



(72) 발명자

맥마이클 로니 딘

미국 워싱턴주 98052-6399 레드몬드 원 마이크로소프트 웨이

루이스 나단 티

미국 워싱턴주 98052-6399 레드몬드 원 마이크로소프트 웨이

슈넬 패트릭

미국 워싱턴주 98052-6399 레드몬드 원 마이크로소프트 웨이

명 세 서

청구범위

청구항 1

치팅(cheating)을 방지하기 위한 방법에 있어서,

신뢰성 있는 구성요소에 의해 장치 및 상기 장치 상에서 실행되는 수정된(modified) 운영체제를 모니터링하는 단계와,

변형 억제 보안 프로세서(tamper resistant security processor) 상에서의 코드의 실행을 포함하는 프록시 실행 동작(proxy execution operation) - 상기 프록시 실행 동작은, 상기 보안 프로세서와 상기 장치의 중앙 처리 장치 사이의 암호로 보호되는(cryptographically protected) 채널을 이용함 - 을 수행하는 단계와,

상기 프록시 실행 동작의 수행과 상기 모니터링의 결과에 기초하여 리소스에 대한 액세스를 제한하는 단계를 포함하는,

치팅 방지 방법.

청구항 2

제 1 항에 있어서,

상기 리소스는 비밀(secret)인 것인,

치팅 방지 방법.

청구항 3

제 1 항에 있어서,

상기 리소스는 네트워크 서비스인 것인,

치팅 방지 방법.

청구항 4

제 1 항에 있어서,

상기 리소스는 추가적인 하드웨어인 것인,

치팅 방지 방법.

청구항 5

제 1 항에 있어서,

상기 신뢰성 있는 구성요소는 신뢰성 있는 플랫폼 모듈인 것인,

치팅 방지 방법.

청구항 6

제 5 항에 있어서,

상기 장치 상에서 실행되는 소프트웨어와 결합된 상기 신뢰성 있는 플랫폼 모듈은 정적인 신뢰 루트 평가(static root of trust measurement)를 생성하는 것인,

치팅 방지 방법.

청구항 7

제 6 항에 있어서,

코드 무결성 동작을 수행하는 단계를 더 포함하는,

치팅 방지 방법.

청구항 8

제 7 항에 있어서,

디스크 무결성 동작을 수행하는 단계를 더 포함하는,

치팅 방지 방법.

청구항 9

제 7 항에 있어서,

개별화(individualization) 메커니즘을 수행하는 단계를 더 포함하는,

치팅 방지 방법.

청구항 10

제 1 항에 있어서,

감시(watchdog) 동작을 수행하는 단계를 더 포함하는,

치팅 방지 방법.

청구항 11

치팅을 방지하기 위한 명령어들을 포함하는 컴퓨터 판독가능 저장 장치에 있어서,

상기 명령어들은,

신뢰성 있는 구성요소에 의해 상기 장치 상에서 실행되는 수정된 운영체제를 모니터링하고,

변형 억제 보안 프로세서 상에서의 코드의 실행을 포함하는 프록시 실행 동작 - 상기 프록시 실행 동작은, 상기 보안 프로세서와 상기 장치의 중앙 처리 장치 사이의 암호로 보호되는 채널을 이용함 - 을 수행하고,

상기 프록시 실행 동작의 수행과 상기 모니터링의 결과에 기초하여 리소스에 대한 액세스를 제한하기 위한 동작을 수행하기 위한 것인,

컴퓨터 판독가능 저장 장치.

청구항 12

제 11 항에 있어서,

상기 리소스는 비밀인 것인,

컴퓨터 판독가능 저장 장치.

청구항 13

제 11 항에 있어서,

상기 리소스는 네트워크 서비스인 것인,

컴퓨터 판독가능 저장 장치.

청구항 14

제 11 항에 있어서,

상기 리소스는 추가적인 하드웨어인 것인,

컴퓨터 판독가능 저장 장치.

청구항 15

치팅을 방지하기 위한 시스템에 있어서,

장치와,

상기 장치 상에서 실행되는 수정된 운영체제와,

변형 억제 보안 프로세서와,

상기 변형 억제 보안 프로세서 상에서의 코드의 실행을 포함하는 프록시 실행 동작을 수행하는 데 이용되고, 상기 보안 프로세서와 상기 장치의 중앙 처리 장치 사이에 존재하는, 암호로 보호되는 채널과,

모듈을 포함하고,

상기 모듈은,

상기 장치와 상기 수정된 운영체제를 모니터링하고,

상기 프록시 실행 동작의 수행과 상기 모니터링의 결과에 기초하여 리소스에 대한 액세스를 제한하도록 구성되는 것인,

치팅 방지 시스템.

청구항 16

제 15 항에 있어서,

상기 리소스는 비밀인 것인,

치팅 방지 시스템.

청구항 17

제 15 항에 있어서,

상기 리소스는 네트워크 서비스인 것인,

치팅 방지 시스템.

청구항 18

제 15 항에 있어서,

상기 리소스는 추가적인 하드웨어인 것인,

치팅 방지 시스템.

청구항 19

제 1 항에 있어서,

상기 장치 상에서 실행되는 제1 운영체제가 상기 수정된 운영 체제를 제공하기 위하여 수정되는 것인,

치팅 방지 방법.

청구항 20

제 11 항에 있어서,

상기 장치 상에서 실행되는 제1 운영체제가 상기 수정된 운영 체제를 제공하기 위하여 수정되는 것인,

컴퓨터 판독가능 저장 장치.

청구항 21

제 15 항에 있어서,

상기 장치 상에서 실행되는 제1 운영체제가 상기 수정된 운영 체제를 제공하기 위하여 수정되는 것인,

치팅 방지 시스템.

발명의 설명

배 경 기 술

[0001] 컴퓨터 게임은 매우 수익성이 좋은 산업이 되었다. 컴퓨터 게임은 단순한 텍스트 기반 게임에서, 복잡하고 움직이는(animated) 그래픽, 음악 및 소리를 포함하는 멀티미디어 실감(immersive) 환경으로 진화하였다. 게임의 쌍방형성 및 소셜 네트워킹 측면을 강화하기 위해, 온라인 환경은 게임 경험의 필수적인 부분이 되어, 게임 팬이 멀티플레이어 게임에 참가하고, 새로운 게임을 다운로드하며, 그들이 보유하는 기존 게임에 새로운 특징을 부가하는 등을 할 수 있도록 한다.

[0002] 온라인 환경은 게이머들이 치팅(cheating)에 결부되는 새로운 기회를 만들기도 하였다. 치팅은, 다른 플레이어에 비해 부당한 이득을 얻기 위한 소프트웨어 보강(augmentation)과 같은 사용자의 여하한 활동을 지칭한다. 멀티플레이어 게임과 같은 어떤 환경에서, 온라인 치팅은 오프라인 치팅보다 더 중요해졌을 수 있다.

[0003] 치팅은 많은 상이한 형태를 가질 수 있다. 가장 간단한 것은 게임내(in-game) 자산에 대한 다른 스펙(예를 들어, 훨씬 빠른 자동차)을 얻거나, 게임 내 환경을 조작하거나, 게임 업적(achievement)을 변경하거나, 다른 플레이어의 저장된 게임의 내용을 바꾸거나 이를 로드하기 위한 로컬 데이터 파일 조작을 포함한다. 물리적 형태를 취할 수도 있다. 웹상에서는, 사람 행동보다 빠르게 인에이블 하는 컨트롤러의 생성 또는 조작을 위한, 래피드 파이어(rapid fire)와 같은 몇몇의 설명(specification)이 있다.

[0004] 이들 치트는, 대중적인 온라인 멀티플레이어 게임에 대해 사용가능한 필터 드라이버 또는 애드온(add-on)과 같은 단순한 소프트웨어 추가의 형태를 취할 수 있다. 이들은, 예를 들어, 헤드업(heads-up) 디스플레이, 자동 맵핑 및 안내 도구(auto-mapping and guiding), 자동 타겟팅(auto-targeting), 자동 마법 걸기(auto-spell casting), 광범위한 매크로 기능(extensive macro capabilities), 보트(bots) 생성까지 다양할 수 있으며, 직접적인 사용자 입력 없이 자동 장치(automaton)로 실행될 수 있다. 예를 들어, 벽이 있는 게임에서 사용자는 벽이 보이지 않게 하거나 자동 타겟팅을 생성하기 위한 "치트"를 발견할 수 있다.

[0005] 치팅은 게임에서 사용자가 불법적으로 성취 또는 보상(awards)을 얻는 것을 지칭할 수도 있다. 성취는 게임 플레이 동안에 제공되는 포상일 수 있으며 게임 플레이에서 명예 뺏지를 나타낼 수 있다. 성취는 오프라인이나 온라인으로 획득될 수 있고, 그러므로 치팅은 둘 중 하나의 모드에서 발생할 수 있다. 그러나, 보통 성취는 온라인으로 보고된다.

[0006] 사용자는, 예를 들어, 그들의 시스템상의 실행파일(executable) 또는 데이터 파일을 추가(augment) 또는 수정함으로써 온라인 치팅에 연루될 수 있다. 치팅은 래피드 파이어를 가능하게 하는 입력 스택에 대한 단순한 조작과 레이스 자동차에 대한 변경을 지칭할 뿐만 아니라, 예를 들어, 헤드업 디스플레이(HUD), 자동 타게팅, 보트 등을 포함하는 다수 사용자 게임에 대해 보이는 복잡한 애드온을 포함할 수도 있다.

[0007] 소프트웨어 치팅은 게임 소프트웨어 발전의 생존에 현저한 경제적 위험을 나타낸다. 온라인 치팅이 만연하게 되면, 게임에 대한 사용자의 관심을 억제하고 그에 따라 게임 판매와 온라인 가입(subscription) 판매 모두에 악영향을 준다.

발명의 내용

과제의 해결 수단

[0008] 앤티 치팅(anti-cheating) 시스템은 수정된 운영 체제와 같은 수정된 환경과, 수정된 환경이 특정 장치에서 실행되고 있음을 확인하기 위한 신뢰성이 있는 외부 엔티티의 결합을 포함할 수 있다. 수정된 환경은, 수정된 환경에 의해 대체되는 원래의 환경에 비해 제한된 환경을 생성하기 위해 특정한 방식으로 수정될 수 있다. 수정된 환경으로의 수정은, 예를 들어, 치팅 또는 바람직하지 않은 사용자 행동을 허용하기 위한 하드웨어 및/또는 소프트웨어에 대한 변경을 검출 및/또는 방지하기 위한 원래 환경에 대한 변경을 포함할 수 있다.

[0009] 일 실시형태에 따르면, 앤티 치팅 시스템은, 앤티 치팅 제한을 포함하는 수정된 운영 체제의 생성과, 수정된 운영 체제가 조작되지 않은 형태로 실행되고 있음을 확인하는데 사용되는 신뢰성이 있는 플랫폼 모듈(trusted platform module ("TPM")) 및 관련되는 정적 신뢰 루트 평가(static root of trust measurement ("SRTM"))의 채용을 통해 구현될 수 있다. TPM의 사용은 보안 해법의 효율성을 돕기 위해 다른 기술과 결합될 수 있다. TPM은 시스템의 나머지의 보안이 코드 무결성 및/또는 디스크 무결성 메커니즘을 통하는 등 다른 메커니즘을 통해 추단될 수 있는 사전결정된 지점까지 평가를 수행할 수 있다.

[0010] 다른 실시형태에 따르면, 수정된 운영 체제가 조작되지 않은 형태로 실행되고 있다는 것의 확인은, 엔티티가 수정된 운영 체제를 평가할 능력을 갖고 그 자체로 신뢰성이 있는 방법을 갖는 한, 여하한 외부의 신뢰성이 있는 엔티티에 의해 구현될 수 있다. 신뢰성이 있는 엔티티 앤티 치팅 메커니즘은 휴대전화(cell phone)과 같은 보안 하드웨어 장치에 의해 구현될 수 있거나, 하이퍼바이저 환경에 의해 소프트웨어에서 구현될 수 있다. 또 다른 실시형태는, 신뢰성이 있는 엔티티가 운영 체제가 존재하지 않는 시스템에서 게임을 실행하는 수정된 실행 환경을 확인하거나 달리 신뢰성이 있는 운영 체제 또는 하이퍼바이저 내의 수정된 에뮬레이션 환경을 확인하는 시스템을 포함한다.

도면의 간단한 설명

[0011] 도 1a는 일 실시형태에 따른 앤티 치팅 시스템을 도시한다.

도 1b는 일 실시형태에 따른 앤티 치팅 프로세스의 동작을 도시하는 흐름도이다.

도 2는 앤티 치팅 기능을 수행하기 위한 신뢰성이 있는 플랫폼 모듈과 정적 신뢰의 루트 모듈의 블록도이다.

도 3a는 일 실시형태에 따른 코드 무결성 동작을 도시한다.

도 3b는 일 실시형태에 따른 디스크 무결성 메커니즘의 동작을 도시한다.

도 4a는 일 실시형태에 따른 프록시 실행 프로세스를 도시한다.

도 4b는 일 실시형태에 따른 감시(watchdog) 프로세스의 동작을 도시한다.

도 4c는 제한된 하드웨어 장치를 통해 앤티 치팅 시스템이 구현되는 것을 도시한다.

도 4d는 일 실시형태에 따른 하이퍼바이저(hypervisor)를 통한 앤티 치팅 시스템의 동작을 도시한다.

도 5는 컴퓨터 형태의 범용 연산 장치를 포함하는 예시적인 실시형태를 구현하기 위한 예시적 시스템을 도시한다.

발명을 실시하기 위한 구체적인 내용

[0012] 도 1a는 일 실시형태에 따른 앤티 치팅 시스템을 도시한다. 일 실시형태에 따르면, 앤티 치팅 시스템(101)은, 클라이언트 장치(110) 상의 수정된 운영 체제(102)와 같은 수정된 환경의 엔지니어링의 결합을 포함할 수 있고, 이는, 수정된 환경, 예를 들어, 수정된 운영 체제(102)가 실제로 클라이언트 장치(110) 상에서 실행되고 있다는 사실을 확인하기 위한 신뢰성이 있는 외부 엔티티(104)의 동작과 연관된다. 수정된 환경, 예를 들어, 수정된 운영 체제(102)는, 수정된 운영 체제(102)에 의해 대체된 원래의 운영 체제(103)와 비교하여 제한된 환경을 생성하기 위한 특정한 방식으로 수정될 수 있다. 일 실시형태에 따르면, 수정된 운영 체제(102)에 있어서의 수정은, 예를 들어, 사용자(105)에 의한 치팅 행동을 방지하기 위한 수정과 같이 치팅을 방지하기 위한 원래 운영 체제(103)에 대한 변경을 포함할 수 있다.

[0013] 수정된 운영 체제(102)는 클라이언트 장치(110) 상에서 고립된 실행 환경(108)을 생성할 수 있다. 고립된 실행 환경(108)은, 모든 소프트웨어가 단일한 공급자(vender)에 의해 직접 서명될 것을 요구하는 것으로부터, 제3자 소프트웨어가 특정 제한 요구를 통과함으로써 인증되고 그에 의해 고립된 실행 환경에서 실행될 수 있게 되는 것이 가능한 좀 더 유연한 시스템을 허용하는 것까지의 스펙트럼일 수 있다. 일 실시형태에 따르면, TPM은 신뢰의 루트(root)일 수 있고 고립된 실행 환경(108)은 코드 측정과 코드가 올바르게 측정된 경우 비밀을 해제하는 것을 통해 주로 수립될 수 있다.

[0014] 고립된 실행 환경(108)은, 클라이언트 장치(110) 상에서의 소프트웨어 실행이 제어되고 식별될 수 있는 환경을 포함할 수 있다. 완전히 고립된 환경에서, 공격자는 클라이언트 장치(110) 상에서 어떤 코드도 실행할 수 없고

더 비용이 높고 번거로운 하드웨어 기반 공격에 의존하여야 한다. 외부 엔티티(104)는, 고립된 실행 환경(108)이 동작하고 있음을, 즉 수정된 운영 체제(102)가 설치되고 실행되고 있음을 확인하기 위한 모니터링 기능을 클라이언트 장치(110) 상에서 수행함으로써, 클라이언트 장치(110) 상에서 고립된 실행 환경(108)이 설치되고 온전하다는 사실을 확인하도록 동작할 수 있다.

[0015] 다시 도 1a를 참조하면, 사용자(105)는 클라이언트 장치(110)를 사용하여 게임 타이틀(112)과 같은 게임이나 기타 엔터테인먼트 소프트웨어를 플레이할 수 있다. 사용자(105)는 온라인과 오프라인 모드 모두에서 클라이언트 장치(110)를 사용할 수 있다. 오프라인이나 온라인 모드에서 게임 타이틀(112)과 상호작용하는 사용자(105)는 성취 보상이나 기타 게임 플레이와 관련되는 리프리젠테이션(representation)을 얻을 수 있다. 성취 보상은, 예를 들어, 획득된 레벨의 리프리젠테이션, 극복한 적의 수 등일 수 있다. 또한, 사용자(105)는 게임 타이틀(112)의 다양한 파라미터를 설정할 수 있는데, 이는 게임 타이틀(112)의 특징을 제어한다. 특징은 난이도 등과 같은 게임 플레이와 관련되는 다양한 기능을 포함할 수 있다.

[0016] 온라인 모드 동안, 사용자(105)는 다른 플레이어(도 1a에 미도시)와 상호작용하여 멀티플레이어 게임 플레이를 허용할 수 있다. 특히, 클라이언트 장치(110)는, 사용자(105)가 멀티플레이어 게임 플레이에서 다른 사용자(미도시)와 상호작용할 수 있도록 하기 위해 네트워크(114)를 통해 엔터테인먼트 서버(116)와 결합될 수 있다. 멀티플레이어 게임 세션 동안, 사용자(105)는 게임 플레이에 영향을 주는 게임 타이틀(112)의 동작과 관련된 다양한 특징을 즐길 수 있다. 이들 특징은 사용자의 플레이어가 무적(invincible)이거나 특정한 불가해성(invulnerability)을 갖는지 여부와 같은 게임 동작을 포함할 수 있다.

[0017] 사용자(105)는, 예를 들어 단일 플레이어 게임 플레이 동안, 오프라인 모드에서 클라이언트 장치(110)와 상호작용할 수도 있다. 후속하여, 사용자(105)는 클라이언트 장치(110)가 온라인으로 되도록 할 수 있다. 오프라인 모드에서 온라인 모드로의 이러한 전이 후에, 오프라인 게임 플레이 동안 사용자(105)가 얻은 다양한 성취는 온라인인 다른 플레이어에게 표시될 수 있다.

[0018] 일 실시형태에 따르면, 기존의 또는 원래 환경, 예를 들어 원래의 운영 시스템(103)은 클라이언트 장치(110)에서 대체될 수 있고, 수정된 운영 시스템(102)에 의한 대체 시에, 이제 게임 타이틀(112)의 실행을 지원할 수 있으나, 제한된 방식으로 지원할 수 있다. 수정된 운영 시스템(102)은, 예를 들어, 엔터테인먼트 또는 게임 소프트웨어를 호스팅하지만 사용자(105)가 특정한 바람직하지않은 치팅 행위에 연루되는 것을 제한하는 운영 체제일 수 있다.

[0019] 일 실시형태에 따르면, 수정된 운영 체제(102)는 게임 타이틀, 예를 들어 112의 실행에 대한 제한된 환경을 생성하는 운영 체제일 수 있다. 즉, 수정된 운영 체제(102)는 사용자(105)의 능력이 소정의 치팅 행위를 수행하는 것을 제한하도록 원래의 운영 체제(103)로부터 제작될 수 있다.

[0020] 예를 들어, 원래의 운영 체제(103)는, 드라이버가 몇몇의 루트 인증 기관(root certificate authorities) 중 하나에 의해 인증된 키에 의해 서명된 이상 여하한 장치 드라이버의 설치를 제한 없이 허용할 수 있다. 수정된 운영 체제(102)는 여하한 장치 드라이버가 중앙 기관(centralized authority)의 특정 키에 의해 서명될 것을 요구함으로써 치팅 행위의 특정한 유형을 방지하도록 수정될 수 있고, 또한 로컬 관리자에 의한 장치 드라이버의 업데이트를 금지할 수 있다.

[0021] 신뢰성이 있는 외부 엔티티(104)는 수정된 운영 체제(102)가 실제로 클라이언트 장치(110) 상에 설치되고 실행되고 있다는 사실을 확인하기 위해 클라이언트(110)에 대해 기능을 수행할 수 있다. 신뢰성이 있는 외부 엔티티(104)는, 신뢰성이 있는 엔티티(104)가 확인하는 엔티티, 즉 수정된 운영 체제(102)보다 높은 정도로 그 동작이 신뢰된다는 점에서 신뢰될 수 있다. 즉, 신뢰성이 있는 엔티티(104)의 동작은 사용자(105)가 실제로 설치된 수정 운영 체제(102)를 갖기 위한 행위보다 훨씬 높은 정도로 신뢰된다.

[0022] 도 1b는 일 실시형태에 따른 앤티 치팅 프로세스의 동작을 도시하는 흐름도이다. 프로세스는 120에서 시작된다. 124에서, 원래의 운영 체제는 고립된 실행 환경을 생성하도록 수정되며, 이는 사용자가 다양한 앤티-치팅 행위에 연루되는 것을 차단하는 제한된 실행 환경일 수 있다. 126에서, 신뢰성이 있는 외부 엔티티는 수정된 운영 체제가 특정 클라이언트 장치상에서 실제로 실행되고 설치되어 있는지 여부를 판정하기 위한 확인(verification)을 수행한다. 프로세스는 128에서 종료한다.

[0023] 도 2는 앤티 치팅 기능을 수행하기 위한 신뢰성이 있는 플랫폼 모듈과 정적 신뢰의 루트 모듈의 블록도이다. 특히, 도 2에 도시된 바와 같이, 도 1a에 도시된 신뢰성이 있는 외부 엔티티의 기능은 TPM(204)에 의해 대체되었다. 일 실시형태에 따르면, TPM(204)은 증명(attestation) 기능 또는 동작을 수행할 수 있다.

[0024] 구체적으로, TPM(204)은 증명 모듈(208)을 통해 증명 기능을 수행할 수 있다. 증명은 정보, 구체적으로 수정된 운영 체제(102)의 클라이언트 장치(110) 상의 설치 및 실행의 정확성을 보증하는 프로세스를 지칭할 수 있다. TPM(204)은, 예를 들어, 부팅 환경과 클라이언트 장치(110)로 로딩되는 구체적인 부팅 사슬(boot chain)을 증명할 수 있다.

[0025] 증명은 클라이언트 장치 상에서 실행되는 소프트웨어에 대한 변화를 허용한다; 그러나, 그 소프트웨어에 대한 그러한 모든 변화는 측정되어 비밀을 공개하거나 네트워크 기능성에 대한 액세스를 허용하는 인증 결정을 허용한다. 증명은 클라이언트 장치(102)와 관련된 하드웨어가 어떤 소프트웨어가 현재 실행 중인지, 예를 들어, 수정된 운영 체제(102)를 명시하는 인증서를 생성하도록 함으로써 이루어질 수 있다. 그러면, 클라이언트 장치(102)는, 그 소프트웨어, 예를 들어, 수정된 운영 체제(120)가 실제로 클라이언트 장치(110) 상에서 온전하게 있으며 조작되지 않았음을 보여주기 위해, 엔터테인먼트 서버(116)와 같은 원격 당사자에게 이 인증서를 제공할 수 있다. 증명은, 꼭 그런 것은 아니지만, 예를 들어 공개 키 암호화와 같은 암호화 기술과 결합될 수 있어, 전송된 정보가 증명을 제시하고 요청한 프로그램에 의해서만 판독될 수 있고 도청자(eavesdropper)에 의해 서는 판독될 수 없다. 이러한 암호화 기술은 사용자의 프라이버시와 시스템에 의해 사용되는 프로토콜의 비밀 모두를 보호할 수 있다.

[0026] 일 실시형태에 따르면, 사용자가 시스템 또는 운영 체제의 특정 버전을 실행하는지 여부를 판정하기 위해 앤티 치팅의 맥락에서 증명이 적용될 수 있다. 즉, 증명은, 클라이언트 장치(110)에서 수정된 운영 체제(102)가 설치되었고 실행되고 있음을 확인하기 위해, TPM(204)에 대해 증명 모듈(208)에 의해 수행될 수 있다. 예를 들어, 사용자(105)가 원래의 운영 체제(103)가 아니라 수정된 운영 체제(102)를 실행하도록 강요하는 것이 바람직할 수 있는데, 수정된 버전이 그 자체로 보안 및 앤티 치팅 보호를 강화하였을 수 있기 때문이다. 즉, 도 1a와 관련하여 설명한 바와 같이, 수정된 운영 체제(102)는 고립된 실행 환경(108)으로 대표되는 제한된 환경을 포함할 수 있다.

[0027] 그러나, 사용자(110)는, 실제로 사용자(110)가 운영 체제의 옛 버전, 즉, 원래의 운영 체제(103)를 실행하는 때에 수정된 운영 체제(102)를 실행하는 척 함으로써 치팅 행위에 연루되려 시도할 수 있다. 이는, 실제로는 구 버전(원래의 운영 체제(103))을 실행하면서 새로운 버전(수정된 운영 체제(102))로부터 특정 비트를 얻음으로써 이루어질 수 있으며, 이에 의해 사용자(105)가 새로운 버전(수정된 운영 체제(102))를 실행하고 있는 것처럼 잘못 표시한다. 이러한 기술은 오늘날 흔한 것으로서, 이에 의해 위반된 DRM 시스템에 대해 패치가 만들어진 후에, 공격자는 패치된 시스템에서 새로운 비밀 정보를 얻고 그 데이터를 예전의 위반된 시스템에 위치시켜 예전의 위반된 시스템이 새로운 패치된 시스템인 것처럼 보이게 하거나, 새로운 패치된 시스템에서의 공격의 동일한 형태 또는 약간 수정된 형태를 새로운 패치된 시스템에 적용함으로써 새로운 위반된 시스템을 생성한다. 모든 경우에, 공격자는 새로운 패치된 DRM 시스템을 실행하고 있지 않다. 공격자가 이들 행위를 수행하는데 있어서의 어려움을 증가시키는 것은 패치를 공개하는데 있어서 작업의 대부분을 나타낸다. 개발자는 새로운 비밀을 추출될 수 있는 용이함과 동일한 종류의 공격이 새로운 패치된 시스템에 적용될 수 있는 용이함을 염두에 두어야 한다. 그러므로, TPM(204)가, 원래의 운영 체제(103)나 원래의 운영 체제(103)와 수정된 운영 체제(120)의 결합이 아니라, 수정된 운영 체제(102)를 사용자가 실행하고 있다는 사실을 증명하도록 하는 것이 바람직할 수 있다. 다르게는, 희망 구성요소는 하드웨어 구성요소일 수 있다. 이 경우, 신뢰성이 있는 구성요소가 사용자가 특정 하드웨어 구성요소를 갖는다는 사실을 증명하는 것이 바람직할 수 있다. 추가적인 대안은, 하드웨어 구성요소가 소프트웨어가 어떤 것을 실행할 수 있는지에 대해 직접적으로 제한을 가하여 환경에 대한 직접 수정을 방지하는 것이다. 다른 대안은 하드웨어 구성요소가 소프트웨어가 로딩할 수 있는 것에 대한 제한을 가하고 로딩한 소프트웨어의 증명을 평가하고 제공하는 것이다. 유사하게, 하이퍼바이저가 이들 모두 또는 하나의 동작을 수행할 수 있다.

[0028] TPM(204)은 정보를 보호하는 암호화 키를 저장할 수 있는 보안 암호 프로세서(secure crypto processor)(도 2에 미도시)를 포함할 수 있다. TPM(204)은, 하드웨어 의사 난수(pseudo-random number) 생성기에 추가하여 암호화 키의 보안 생성과 그 사용에 대한 제한을 위한 장치(facilities)제공할 수 있다. 또한, 설명되는 바와 같이, TPM(204)은 증명 및 봉인 저장(sealed storage)과 같은 기능도 제공할 수 있다. 증명은, 클라이언트 장치(110)의 하드웨어 및 소프트웨어 구성(예를 들어, 시스템(102)과 애플리케이션(112))와 같은 하드웨어와 소프트웨어 구성의, 또는 하드웨어 구성 단독의, 또는 소프트웨어 구성 단독의 거의 복제 불가한 해시 키 요약을 생성하는 TMP(204)에 의해 이루어질 수 있다. 일 실시형태에 따르면, 복제 불가한 키 요약(들)은 플랫폼 구성 레지스터("PCR")(206)에 저장될 수 있다.

[0029] 그러면 엔터테인먼트 서버(116)와 같은 제3자는 수정된 운영 체제(204)와 같은 소프트웨어가 클라이언트 장치

(110) 상에서 온전하고 변화되지 않았음을 그 증명을 이용하여, 예를 들어 도 1a에 대하여 설명한 바와 같이 확인할 수 있다. TPM(204)은 그 후 봉인 프로세스를 수행할 수 있는데, 이는 데이터를 TPM(204)이 관련 복호화 키를 공개하였을 경우에만 복호화할 수 있는 방식으로 암호화하며, TPM(204)은 TPM(204)의 PCR로의 평가에 따라 올바른 아이덴티티를 가진 소프트웨어에 대해서만 복호화 키를 공개한다. 봉인 프로세스 동안 어떤 소프트웨어가 비밀을 봉인해제할 수 있는지를 특정함으로써, 엔터테인먼트 서버는 이들 비밀의 오프라인 사용을 허용하면서도 보호를 유지할 수 있다.

[0030] STRM(209)는 강력한 평가를 제공하는 반면, 현대의 운영 체제에서는 개별 바이너리의 개수를 다루는 것이 번잡하게 될 수 있다. 이 이슈를 다루기 위해, TPM(204)은, 시스템의 나머지의 보안 및 무결성이 코드 무결성 및 디스크 무결성 또는 다른 무결성 기반 메커니즘(후술)을 이용하는 등의 다른 메커니즘을 통해 관리될 수 있는 사전결정된 지점까지 평가를 수행할 수 있다.

[0031] 도 3a는 일 실시형태에 따른 코드 무결성 동작을 도시한다. 코드 무결성은 로딩되는 여하한 바이너리도 신뢰성이 있는 루트 기관에 의해 암호화적으로(cryptographically) 서명될 것을 요구할 수 있다. 그러므로, 도 3a에 도시된 바와 같이, OS 커널(302)뿐만 아니라 바이너리 파일(304(1)-304(N))은 암호화적으로 서명되는데, 즉, 암호화적으로 보안인 환경(315)에서 존재한다. 다양한 파일(304(1)-304(N) 및 302)이 사용되기 위해, 이들은 신뢰성이 있는 루트 기관(306)에 의해 확인되어야 한다.

[0032] 예를 들어, 일부 운영 체제는 커널 모드 구성요소를 위한 코드 무결성 메커니즘을 포함할 수 있다. 일 실시형태에 따르면, 이 기존 메커니즘은 책임 있는 구성요소 (및 그 지점까지의 부팅 경로의 내용)를 평가하고 그 후 운영 체제 커널의 보안을 추단(infer)함으로써 레버리지될 수 있다. 일 실시형태에 따르면, 바이너리 서명은 로드할 수 있는 드라이버의 세트를 제한하도록 새로운 키로 루트될 수 있다. 또한, 사용자 모드 바이너리가 동일한 서명을 보유할 것을 요구하는 확장(extension)이 이 시스템에 접목될 수 있다. 코드 무결성의 사용은, 그럼에도, 레지스트리 또는 기타 구성 파일과 같은 보안에 필수적인 비-바이너리 파일의 세트를 여전히 남길 수 있다.

[0033] 일 실시형태에 따르면, 디스크 무결성 메커니즘은 이들 나머지 홀을 패치하는 방법을 제공할 수 있다. 도 3b는 일 실시형태에 따라 디스크 무결성 메커니즘의 동작을 도시한다. 디스크 무결성은, 하드 디스크와 같은 지속적인 매체의 제어를 통해 고립 실행 환경을 보장하려고 할 수 있다. 그러므로, 도 3b에 도시된 바와 같이, 환경(324) 내의 무결성 보호 디스크 파일(320(1)-320(N)은 사용될 수 있기 전에 모듈(322)을 이용하여 무결성 체크될 수 있다. 디스크로부터 판독된 데이터가 암호화적으로 보호되는 것(예를 들어, 디지털 서명되는 것)을 보장함으로써, 공격 소프트웨어를 시스템으로 주입하는 것의 어려움이 증가될 수 있고, 그에 의해 그것이 일단 수립되면 공격을 지속하는 것이 특히 어렵게 된다.

[0034] 디스크 무결성 모델에 따르면, 파티션 상의 파일, 예를 들어, 320(1)-320(N)은 루트 키에 의해 서명될 수 있다. 이는 디스크가 효과적으로 판독 전용일 것을 요구할 수 있다 (파일은 클라이언트 상에서 서명될 수 없으므로). 디스크 무결성 모델은 데이터뿐만 아니라 코드가 서명되는 것을 보장함으로써 코드 무결성의 취약성(vulnerability)을 다룰 수 있다.

[0035] 다른 실시형태에 따르면, 프록시 실행 메커니즘이 채용될 수 있다. 프록시 실행은 메인 중앙 처리 유닛("CPU") 외의 다른 곳에서 코드를 실행하는 것을 지칭할 수 있다. 예를 들어, 코드는 보안 프로세서 상에서 실행될 수 있고, 보안 프로세서는 메인 CPU에 비해 조작 및 스누핑(snooping)에 더 저항적일 수 있다. 이러한 방식으로, 코드의 특정 부분의 실행을 다양한 방식으로 보안 프로세서로 옮김으로써 하드웨어가 레버리지될 수 있다.

[0036] 일 실시형태에 따르면, 프록시 실행은 CPU와 보안 프로세서 사이의 채널이 라이센스될 것을 요구함으로써 더 강화될 수 있다. 구체적으로, 도 4a는 일 실시형태에 따른 프록시 실행 프로세스를 도시한다. 코드(406)는 보안 프로세서(404) 상에서 실행될 수 있다. 보안 프로세서(404)는 라이센스된 채널(410)을 통해 CPU(402)와 통신할 수 있다.

[0037] 일 실시형태에 따르면, 프록시 실행 메커니즘은 보안 프로세서를 특정 SRTM 코드 평가에 연결시키는데 사용될 수 있고, 라이센스가 주기적으로 만료하는 경우에는, 복구의 형식으로서 시스템의 재확인을 강제하는데 사용될 수 있다. 이 실시형태에 따르면, 보안 프로세서와 CPU 사이의 모든 통신(채널을 수립하는데 필요한 통신 제외)은 세션 키에 대한 지식을 요구할 수 있다. 이를 수립하기 위해, 보안 프로세서는, 그러한 기능을 가지고 있다면 TPM으로부터의 증명을 직접 확인하거나, 기능을 가지지 않거나 더 유연한 버저닝(versioning)이 요구된다면, 엔터테인먼트 서버(116)와 같은 신뢰성이 있는 제3자가 TPM과 보안 프로세서 사이에서 협상하는데 사용될

수 있고, 이에 의해 엔터테인먼트 서버는 TPM으로부터의 증명을 확인하고 보안 프로세서와 TPM 사이의 암호화적으로 보호되는 채널의 수립을 허용하기 위한 증거(proof)와 중요 내용(key material)을 제공한다. 그러면 TPM은, TPM이 인증된 PCR 값에 대한 모든 비밀을 보호하는 것과 동일한 방식으로 이후의 장치의 재부팅 상에서 CPU로 이들 비밀을 제공할 수 있다.

[0038] 다른 실시형태에 따르면, 개별화(individualization) 메커니즘이 채용될 수도 있다. 개별화는 단일 머신 또는 사용자에게 향하는 코드를 구축하고 전달하는 프로세스를 지칭할 수 있다. 예를 들어, 애플리케이션의 동작에 필수적인 코드는 서버상에서 구축(온-디맨드로, 또는 미리 풀(pool)의 형태로)되고 활성화 프로세스의 일부로서 클라이언트로 다운로드될 수 있다.

[0039] 일 실시형태에 따르면, 개별화는 공격자가 해적 컨텐트를 사용하고자 하는 각 머신에 대해 실질적인 작업을 수행하도록 함으로써 "Break Once Run Everywhere" 위반을 감소시키려고 할 수 있다. 개별화 프로세스에 의해 생성된 바이너리는 고유하므로 (또는 적어도 "대부분 고유하므로(mostly unique)"), 모든 머신을 위태롭게 할 수 있는 단순한 패치를 배포하는 것이 더 어렵게 된다.

[0040] 개별화는 일부 형태의 강한 아이덴티티를 요구할 수 있는데, 머신 바인딩에 대해 이는 하드웨어 식별자가 된다. 전통적인 DRM 시스템에서, 이는 시스템 내 다양한 하드웨어의 ID를 결합함으로써 도출되었다. 일 실시형태에 따르면, 고유하고 강력한 하드웨어 식별자를 제공하기 위해 보안 프로세서가 레버리지될 수 있다.

[0041] 열악한 고립 환경을 갖는 시스템과 같은 특정 시스템에서, 치팅 (및 더 약한 정도에서는 불법복제(piracy))를 검출하기 위한 방법은 감시(watchdog) 프로세스를 채용할 수 있고, 이는 게임 서버(116)로부터의 도전에 응답할 수 있다. 게임 서버(116)에 의해 수집된 데이터는 이후에 머신이 침해(compromised)되었는지 여부를 판정하는데 사용될 수 있고, 침해되었다면 머신은 게임 서버(116)로부터 금지되고 내용에 대한 추가적인 라이센스를 획득하는 것이 차단될 수 있다.

[0042] 일 실시형태에서, 감시 프로세스는 그 자체로, 상술한 이전 실시형태에서 설명된 바와 같이 운영 체제 또는 실행 환경을 보호하는 동일한 메커니즘에 의해 무결성 보호될 수 있다.

[0043] 도 4b는 일 실시형태에 따른 감시 프로세스의 동작을 도시한다. 도 4b에 도시된 바와 같이, 앤티 치팅 시스템(101)은 클라이언트 장치(110) 상의 감시 프로세스(430)를 포함하도록 더 구성되었다. 엔터테인먼트 서버(116)는, 네트워크(114)를 통해 전송되고 감시 프로세스(430)에 의해 수신되는 보안 도전(challenge)(432)을 생성할 수 있다. 다음으로, 감시 프로세스(430)는 응답(434)을 생성할 수 있고, 감시 프로세스(430)는 이를 네트워크(14)를 통해 엔터테인먼트 서버(116)로 전송할 수 있다. 수정된 운영 체제(102) 내에서의 실행에 추가하여, 감시 프로세스(430)는 하이퍼바이저에서도 실행될 수 있다.

[0044] 감시 프로세스(430)의 구현은 몇몇 기준의 시행을 요구할 수 있다. 먼저, 이들 도전에 대한 응답의 코드북(codebook)을 사소하게 생성하는 것이 쉬워서는 안되며, 이는 도전의 세트는 커야하고 올바른 답은 클라이언트에게 명확하게 사용가능해서는 안 된다는 것을 의미한다. 감시 프로세스(430)는 새로운 도전이 뜻에 따라 추가될 수 있도록 충분한 민첩성을 요구할 수 있다. 최악의 경우에, 공격자는 인터넷 어딘가에서 깨끗한 시스템을 설정할 수 있는데, 공격 소프트웨어를 실행하는 클라이언트의 시스템은 이를 오라클(oracle)로서 이용할 수 있다. 이를 방지하기 위해, 게임 서버(116)와 감시 프로세스(430) 사이의 채널에 (트리비얼 스누핑(trivial snooping)과 중간자(man in the middle) 공격을 방지하기 위해) 암호화가 적용될 수 있으며, 잠재적으로 응답은 보안 프로세서, 머신 아이덴티티, TPM과 같은 다른 보안에 어떤 방식으로 연결될 수 있다.

[0045] 계층화된 보호 방식도 채용될 수 있는데, 여기서는 다수의 기술이 다양한 결합으로 계층화된다. 예를 들어, 감소된 공격 족적(footprint), 디스크 무결성 및 SRTM을 사용하는 "듀얼 부팅(dual boot)" 방식이 채용될 수 있다.

[0046] 다른 실시형태에 따르면, 이전에 설명된 TPM일 수 있는 도 1a에 도시된 바와 같은 신뢰성이 있는 외부 엔티티(104)를 사용하기보다, 신뢰성이 있는 엔티티는, 하드웨어 장치가 수정된 운영 체제(102)가 실제로 그 하드웨어 장치에서 실행되고 있다는 것을 강제할 수 있는 한, 수정된 운영 체제(102)가 호스팅되는 바로 그 하드웨어 장치를 포함할 수 있다. 그러므로, 도 4c에 도시된 바와 같이, 앤티 치팅 시스템(101)은 제한된 하드웨어 장치(430)를 통해 구현되는데, 이는 수정된 운영 체제(102)와 같은 희망되는 운영 체제의 동작 및 실행을 내부적으로 강제할 수 있다. 제한된 하드웨어 장치(430)는, 예를 들어, 셀룰러 전화 또는 케이블 셋 탑 박스일 수 있다.

[0047] 또 다른 실시형태에 따르면, 신뢰성이 있는 외부 엔티티(104)를 채용하는 것보다, 하이퍼바이저(440)는 내부적으로 클라이언트 장치(110)에 제공될 수 있다. 도 4d는 일 실시형태에 따른 하이퍼바이저(hypervisor)를 통한

앤티 치팅 시스템의 동작을 도시한다. 하이퍼바이저(440)는 이전에 설명된 바와 같이 TPM 환경에서 SRTM을 채용하는 대신에 평가의 부담을 받을 수 있다. 일 실시형태에 따르면, 하이퍼바이저(440)는 평가된 부팅 하이퍼바이저일 수 있다. 이 실시형태에 따르면, 실행되고 있는 특정 하이퍼바이저(440)의 아이덴티티를, 예를 들어, 하이퍼바이저 아이덴티티 모듈(442)을 통해 판단하기 위해 메커니즘이 제공될 수 있다. 그러므로, 예를 들어, 엔터테인먼트 서버(116)는 하이퍼바이저(440)의 본질을 식별하기 위해 하이퍼바이저 아이덴티티 모듈과 통신할 수 있다. 다음으로, 하이퍼바이저(440)는, 엔터테인먼트 서버(116)에 수정된 운영 체제(102)의 존재 또는 부재를 통지하기 위해 엔터테인먼트 서버(116)와 통신할 수 있다. 수정된 운영 체제(102)의 설치는 게임 타이틀(112)의 동작으로의 비-치팅(non-cheating) 맥락을 보장한다.

[0048] 도 5는 예시적인 실시형태의 측면이 구현될 수 있는 예시적인 연산 환경을 도시한다. 연산 시스템 환경(500)은 적당한 연산 환경의 단지 하나의 예이고, 설명된 예시적 실시형태의 사용 또는 기능성의 범위에 대한 여하한 제한도 제시하려는 것이 아니다. 연산 환경(500)은, 예시적인 연산 환경(500)에서 설명된 구성요소 중 여하한 하나 또는 그들의 결합과 관련하여 어떠한 의존성이나 요구사항을 갖는 것으로 해석되어서도 안 된다.

[0049] 예시적인 실시형태는, 많은 다른 범용 또는 전용 연산 시스템 환경 또는 구성과 함께 동작할 수 있다. 예시적 실시형태와 함께 사용하기에 적합할 수 있는 공지된 연산 시스템, 환경 및/또는 구성의 예는 퍼스널 컴퓨터, 서버 컴퓨터, 핸드-헬드 또는 랩탑 장치, 멀티프로세서 시스템, 마이크로프로세서 기반 시스템, 셋탑 박스, 프로그램가능 소비자 가전제품(consumer electronics), 네트워크 PC, 미니컴퓨터, 메인프레임 컴퓨터, 임베디드 시스템, 위의 시스템 또는 장치 중 여하한 것을 포함하는 분산 컴퓨팅 환경 등을 포함하지만 이에 제한되지 않는다.

[0050] 예시적인 실시형태는, 컴퓨터에 의해 실행되는 프로그램 모듈과 같은 컴퓨터 실행가능 명령어의 일반적인 맥락에서 설명될 수 있다. 일반적으로, 프로그램 모듈은, 특정 태스크를 수행하거나 특정 추상 데이터 유형을 구현하는 루틴, 프로그램, 객체(object), 구성요소(component), 데이터 구조 등을 포함한다. 예시적인 실시형태는, 태스크가 통신 네트워크 또는 기타 데이터 송신 매체를 통해 링크된 원격 처리 장치들에 의해 수행되는 분산 컴퓨팅 환경에서 실행될 수도 있다. 분산 컴퓨팅 환경에서, 프로그램 모듈 및 기타 데이터는 메모리 저장 장치를 포함하는 로컬 및 원격 컴퓨터 저장 매체 모두에 위치될 수 있다.

[0051] 도 5를 참조하면, 예시적인 실시형태를 구현하기 위한 예시적인 시스템은 컴퓨터(510)의 형태로 범용 연산 장치를 포함한다. 컴퓨터(510)의 구성요소는, 처리 유닛(520), 시스템 메모리(530) 및 시스템 메모리를 포함하는 다양한 시스템 구성요소를 처리 유닛(520)에 결합하는 시스템 버스(521)를 포함할 수 있지만 이에 제한되지 않는다. 처리 유닛(520)은 멀티스레드(multithreaded) 프로세서에서 지원되는 것과 같은 다수의 논리 처리 유닛을 나타낼 수 있다. 시스템 버스(521)는 메모리 버스 또는 메모리 제어기, 주변(peripheral) 버스, 및 다양한 버스 구조 중 임의의 것을 이용하는 로컬 버스를 포함하는 몇몇 버스 구조 유형 중 어떤 것일 수 있다. 제한이 아니라 예로서, 그러한 구조는 ISA(Industry Standard Architecture) 버스, MCA(Micro Channel Architecture) 버스, EISA(Enhanced ISA) 버스, VESA(Video Electronics Standards Association) 로컬 버스 및 PCI(Peripheral Component Interconnect) 버스(메자닌(Mezzanine) 버스라고도 알려져 있음)을 포함한다. 시스템 버스(521)는, 통신하는 장치 사이의, 점대점(point-to-point) 접속, 스위칭 패브릭 등으로 구현될 수도 있다.

[0052] 컴퓨터(510)는 통상 다양한 컴퓨터 판독가능 매체를 포함한다. 컴퓨터 판독가능 매체는 컴퓨터(510)에 의해 액세스될 수 있는 여하한 사용가능한 매체일 수 있고, 휘발성 및 비휘발성 매체, 이동식(removable) 및 비 이동식(non-removable) 매체 모두를 포함한다. 제한이 아니라 예시로서, 컴퓨터 판독가능 매체는 컴퓨터 저장 매체 및 통신 매체를 포함할 수 있다. 컴퓨터 저장 매체는, 컴퓨터 판독가능 명령어들, 데이터 구조, 프로그램 모듈 또는 기타 데이터와 같은 정보를 저장하기 위한 여하한 방법 또는 기술에서 구현되는 휘발성 및 비휘발성, 이동식 및 비 이동식 매체를 포함한다. 컴퓨터 저장 매체는 RAM, ROM, EEPROM, 플래시 메모리 또는 기타 메모리 기술, CDROM, DVD(digital versatile disks) 또는 기타 광 디스크 저장소, 자기 카세트, 자기 테이프, 자기 디스크 저장소 또는 기타 자기 저장 장치, 또는 희망 정보를 저장하는데 사용될 수 있고 컴퓨터(510)에 의해 액세스될 수 있는 여하한 기타 매체를 포함하지만 이에 제한되지 않는다. 통신 매체는 통상 컴퓨터 판독가능 명령어들, 데이터 구조, 프로그램 모듈 또는 기타 데이터를 반송파(carrier wave)와 같은 변조된 데이터 신호나 기타 전송 메커니즘에 포함하며, 여하한 정보 전달 매체를 포함한다. "변조된 데이터 신호"라는 용어는, 신호 내에 정보를 인코딩하기 위한 방식으로 그 특성 세트 중 하나 이상을 갖거나 변화된 신호를 의미한다. 제한이 아니라 예시로서, 통신 매체는 유선 네트워크 또는 직접 유선 접속과 같은 유선 매체 및 음향, RF, 적외선 및 기타 무선 매체와 같은 무선 매체를 포함한다. 상기 중 여하한 것의 결합도 컴퓨터 판독가능 매체의 범위에 포함되

어야 한다.

[0053] 시스템 메모리(530)는 ROM(read only memory)(531) 및 RAM(random access memory)(532)과 같은 휘발성 및/또는 비휘발성 메모리의 형태로 컴퓨터 저장 매체를 포함한다. 시동 동안 등에서, 컴퓨터(510) 내의 소자들 사이에서 정보를 전달하는 것을 돕는 기본적 루틴을 포함하는 기본 입력/출력 시스템(533)(BIOS)은 통상 ROM(531)에 저장된다. RAM(532)은 통상 처리 유닛(520)에 의해 즉시 액세스가능하거나 및/또는 현재 동작되고 있는 데이터 및/또는 프로그램 모듈을 포함한다. 제한이 아니라 예시로서, 도 5는 운영 체제(534), 애플리케이션 프로그램(535), 기타 프로그램 모듈(536) 및 프로그램 데이터(537)를 도시한다.

[0054] 컴퓨터(510)는 기타 이동식/비 이동식, 휘발성/비휘발성 컴퓨터 저장 매체를 포함할 수도 있다. 예시만을 위하여, 도 6은 비 이동식, 비휘발성 자기 매체로부터 판독하고 그에 기록하는 하드 디스크 드라이브(540), 이동식, 비휘발성 자기 디스크(552)로부터 판독하고 그에 기록하는 자기 디스크 드라이브(551), 및 CD ROM 또는 기타 광 매체와 같은 이동식, 비휘발성 광 디스크(556)으로부터 판독하고 그에 기록하는 광 디스크 드라이브(555)를 도시한다. 예시적인 운영 환경에서 사용될 수 있는 다른 이동식/비 이동식, 휘발성/비휘발성 컴퓨터 저장 매체는 자기 테이프 카세트, 플래시 메모리 카드, DVD, 디지털 비디오 테이프, 고체상태(solid state) RAM, 고체상태 ROM 등을 포함하지만 이에 제한되지 않는다. 하드 디스크 드라이브(541)는 통상 인터페이스(540)와 같은 비 이동식 메모리 인터페이스를 통해 시스템 버스(521)에 접속되고, 자기 디스크 드라이브(551) 및 광 디스크 드라이브(555)는 통상 인터페이스(550)와 같은 이동식 메모리 인터페이스에 의해 시스템 버스(521)에 접속된다.

[0055] 위에서 논의되고 도 5에 도시된 드라이브 및 그와 관련된 컴퓨터 저장 매체는 컴퓨터 판독가능 명령어들, 데이터 구조, 프로그램 모듈 및 기타 컴퓨터(510)를 위한 데이터의 저장을 제공한다. 도 5에서, 예를 들어, 하드 디스크 드라이브(541)는 운영 체제(544), 애플리케이션 프로그램(545), 기타 프로그램 모듈(546) 및 프로그램 데이터(547)를 저장하는 것으로 도시된다. 이들 요소는 운영 체제(534), 애플리케이션 프로그램(535), 기타 프로그램 모듈(536) 및 프로그램 데이터(537)와 다를 수도 있고 같을 수도 있음을 유의하라. 운영 체제(544), 애플리케이션 프로그램(545), 기타 프로그램 모듈(546)과 프로그램 데이터(547)는 여기서, 최소한 그들이 다른 복사본임을 나타내기 위해 다른 숫자가 주어진다. 사용자는 키보드(562)와, 보통 마우스, 트랙볼 또는 터치 패드라고 불리는 포인팅 장치(561)와 같은 입력 장치를 통해 컴퓨터(510)로 코맨드들 및 정보를 입력할 수 있다. 다른 입력 장치(미도시)는 마이크로폰, 조이스틱, 게임 패드, 위성 접시, 스캐너 등을 포함할 수 있다. 이들 및 다른 입력 장치는 보통 시스템 버스에 결합된 사용자 입력 인터페이스(560)를 통해 처리 유닛(520)으로 접속되지만, 병렬 포트, 게임 포트 또는 USB(universal serial bus)와 같은 다른 인터페이스 및 버스 구조에 의해 접속될 수 있다. 모니터(591)나 기타 유형의 디스플레이 장치도, 비디오 인터페이스(590)와 같은 인터페이스를 통해 시스템 버스(521)에 접속된다. 모니터에 추가하여, 컴퓨터는 스피커(597)와 프린터(596)와 같은 다른 주변 출력 장치도 포함할 수 있는데, 이는 출력 주변 인터페이스(595)를 통해 접속될 수 있다.

[0056] 컴퓨터(510)는 원격 컴퓨터(580)와 같은 하나 이상의 원격 컴퓨터로의 논리적 접속을 이용하여 네트워크 환경에서 동작할 수 있다. 원격 컴퓨터(580)는 퍼스널 컴퓨터, 서버, 라우터, 네트워크 PC, 피어 디바이스(peer device) 또는 기타 통상의 네트워크 노드일 수 있고, 도 5에 메모리 저장 장치(581)만이 도시되었지만, 보통 컴퓨터(510)와 관련하여 상술한 요소 중 많은 것 또는 전부를 포함한다. 도 5에 도시된 논리적 접속은 LAN(local area network)(571) 및 WAN(wide area network)(573)을 포함하지만, 다른 네트워크도 포함할 수 있다. 이러한 네트워크 환경은 사무실, 전사적(enterprise-wide) 컴퓨터 네트워크, 인트라넷 및 인터넷에서 흔하다.

[0057] LAN 네트워크 환경에서 사용되는 때에, 컴퓨터(510)는 네트워크 인터페이스 또는 어댑터(570)를 통해 LAN(571)에 접속된다. WAN 네트워크 환경에서 사용되는 때에, 컴퓨터(510)는 보통 모뎀(572)이나, 인터넷과 같은 WAN(573)을 통해 통신을 수립하기 위한 다른 수단을 포함한다. 내장 또는 외장일 수 있는 모뎀(572)은 사용자 입력 인터페이스(560) 또는 기타 적당한 메커니즘을 통해 시스템 버스(521)에 접속될 수 있다. 네트워크 환경에서, 컴퓨터(510)에 대해 도시된 프로그램 모듈 또는 그 일부는 원격 메모리 저장 장치에 저장될 수 있다. 제한이 아니라 예로서, 도 5는 원격 애플리케이션 프로그램(585)이 메모리 장치(581)에 상주하는 것으로 도시한다. 도시된 네트워크 접속은 예시적인 것이고 컴퓨터 사이에 통신 링크를 수립하는 다른 수단이 사용될 수 있음을 인식할 것이다.

[0058] 연산 환경(500)은 보통 적어도 소정 형태의 컴퓨터 판독가능 매체를 포함한다. 컴퓨터 판독가능 매체는 연산 환경(500)에 의해 액세스될 수 있는 여하한 사용가능한 매체일 수 있다. 제한이 아니라 예로서, 컴퓨터 판독가능 매체는 컴퓨터 저장 매체와 통신 매체를 포함할 수 있다. 컴퓨터 저장 매체는, 컴퓨터 판독가능 명령어들, 데이터 구조, 프로그램 모듈 또는 기타 데이터와 같은 정보를 저장하기 위한 여하한 방법 또는 기술에서 구현되

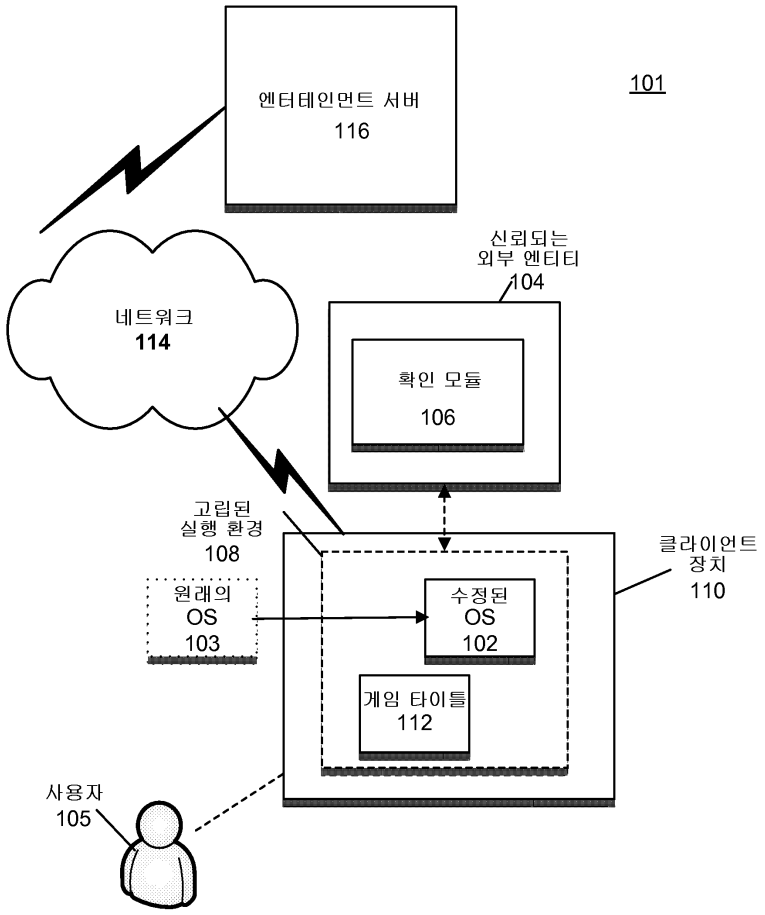
는 휘발성 및 비휘발성, 이동식 및 비 이동식 매체를 포함한다. 컴퓨터 저장 매체는 RAM, ROM, EEPROM, 플래시 메모리 또는 기타 메모리 기술, CDROM, DVD(digital versatile disks) 또는 기타 광 저장소, 자기 카세트, 자기 테이프, 자기 디스크 저장소 또는 기타 자기 저장 장치, 또는 희망 정보를 저장하는데 사용될 수 있고 연산 환경(500)에 의해 액세스될 수 있는 여하한 기타 매체를 포함하지만 이에 제한되지 않는다. 통신 매체는 통상 컴퓨터 판독가능 명령어들, 데이터 구조, 프로그램 모듈 또는 기타 데이터를 반송파(carrier wave)와 같은 변조된 데이터 신호나 기타 전송 메커니즘에 포함하며, 여하한 정보 전달 매체를 포함한다. "변조된 데이터 신호"라는 용어는, 신호 내에 정보를 인코딩하기 위한 방식으로 그 특성 세트 중 하나 이상을 갖거나 변화된 신호를 의미한다. 제한이 아니라 예시로서, 통신 매체는 유선 네트워크 또는 직접 유선 접속과 같은 유선 매체 및 음향, RF, 적외선 및 기타 무선 매체와 같은 무선 매체를 포함한다. 상기 중 여하한 것의 결합도 컴퓨터 판독가능 매체의 범위에 포함되어야 한다. 청구대상이 구조적 특징 및/또는 방법론적 행위에 특정한 언어로 설명되었지만, 첨부된 청구범위에서 정의된 청구대상은 상술된 구체적인 특징 또는 행위에 제한되어야만 하는 것이 아님을 이해하여야 한다. 오히려, 상술한 구체적인 특징 및 행위는 청구항을 구현하는 예시적인 형태로 개시된다.

[0059] 청구대상이 구조적 특징 및/또는 방법론적 행위에 특정한 언어로 설명되었지만, 첨부된 청구범위에서 정의된 청구대상은 상술된 구체적인 특징 또는 행위에 제한되어야만 하는 것이 아님을 이해하여야 한다. 오히려, 상술한 구체적인 특징 및 행위는 청구항을 구현하는 예시적인 형태로 개시된다.

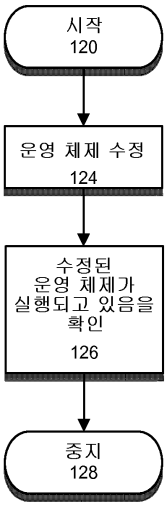
[0060] 본 발명의 청구대상은 법률적 요건을 만족시키는 구체성으로 설명된다. 그러나, 설명 그 자체는 본 특허의 범위를 제한하려는 것이 아니다. 오히려, 청구된 청구대상은 다른 방식으로도 구현되어 본 문서에서 설명된 것과 유사한 단계의 결합이나 다른 단계들을 포함하고, 다른 현재 또는 미래의 기술과 결합될 수 있음이 고려된다.

도면

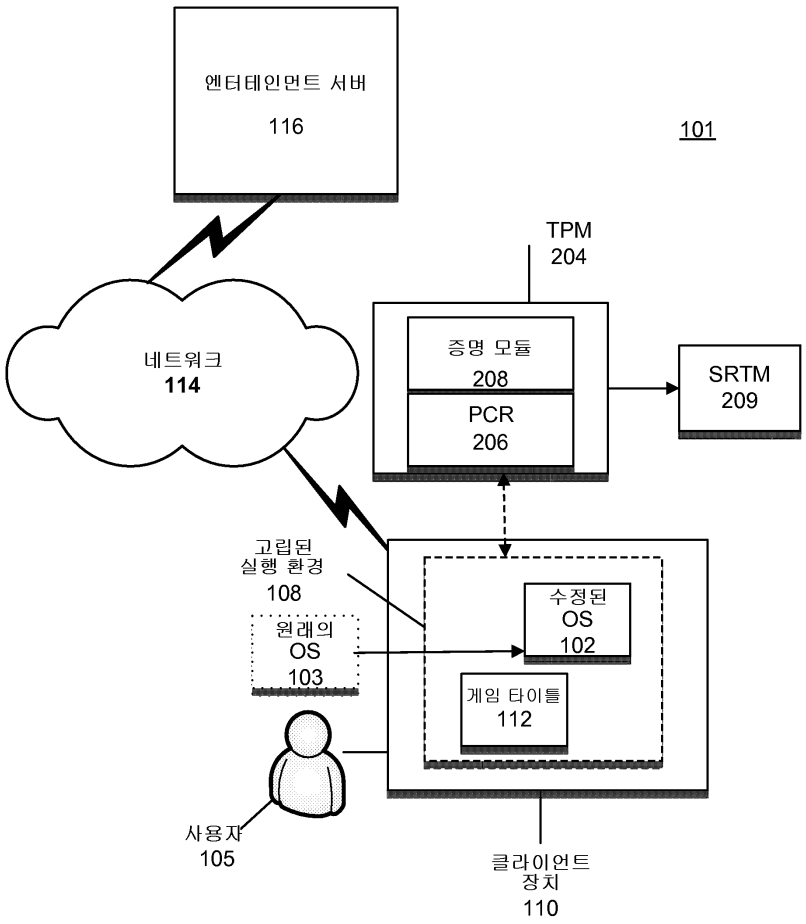
도면1a



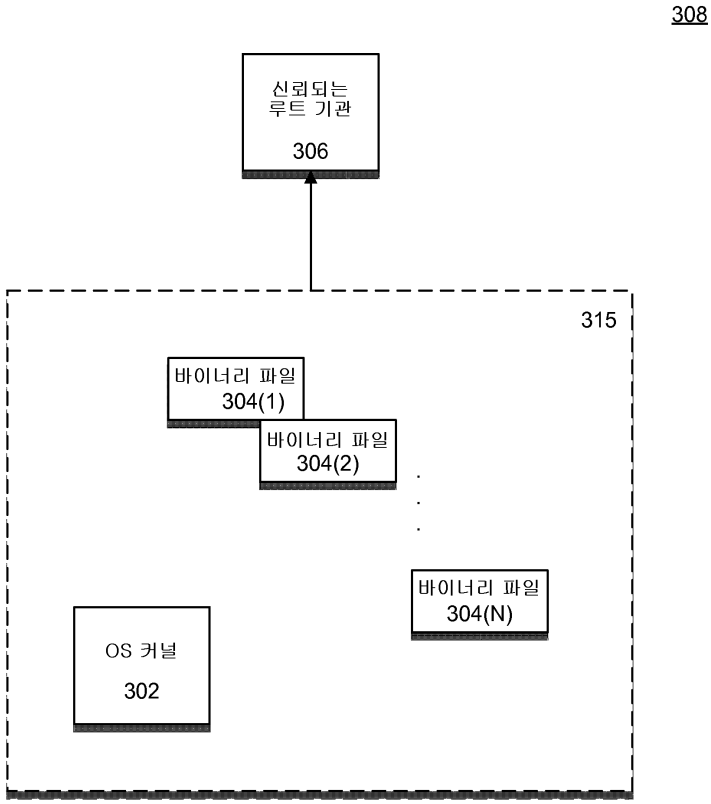
도면1b


시작
120
운영 체제 수정
124
수정된
운영 체제가
실행되고 있음을
확인
126
중지
128

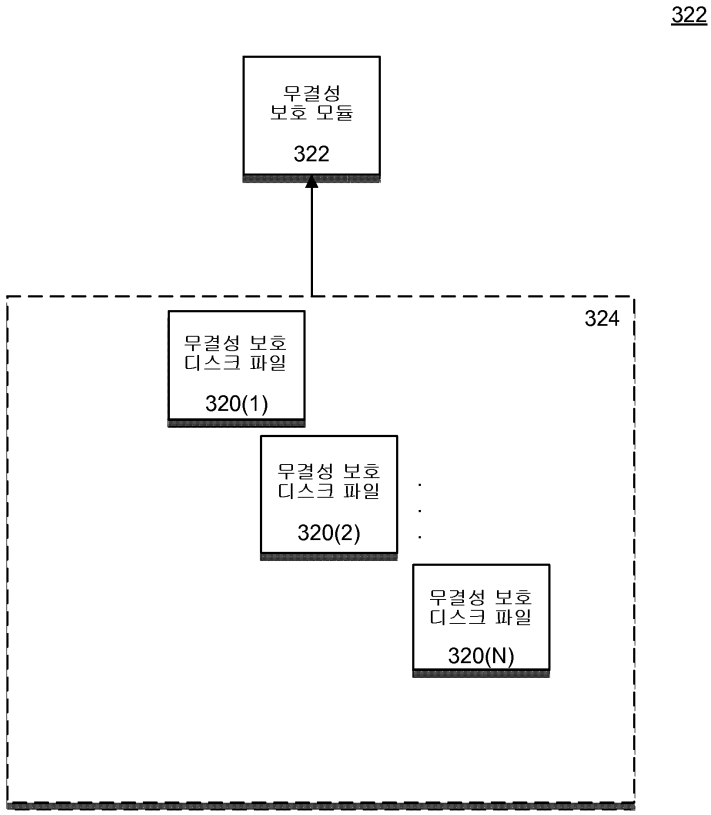
도면2


엔터테인먼트 서버
116
101
TPM
204
네트워크
114
증명 모듈
208
PCR
206
SRTM
209
고립된
실행 환경
108
원래의
OS
103
수정된
OS
102
게임 타이틀
112
사용자
105
클라이언트
장치
110

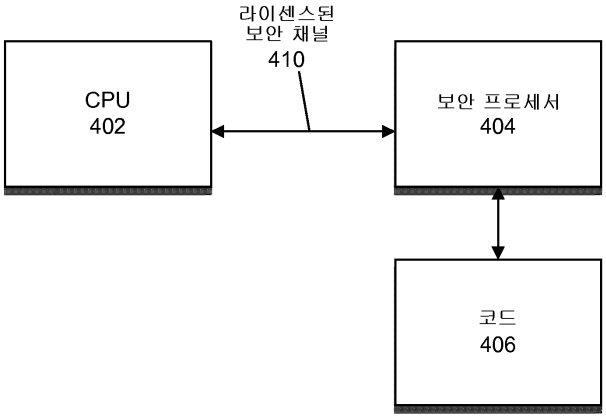
도면3a


308
신뢰되는
루트 기관
306
315
바이너리 파일
304(1)
바이너리 파일
304(2)
바이너리 파일
304(N)
OS 커널
302

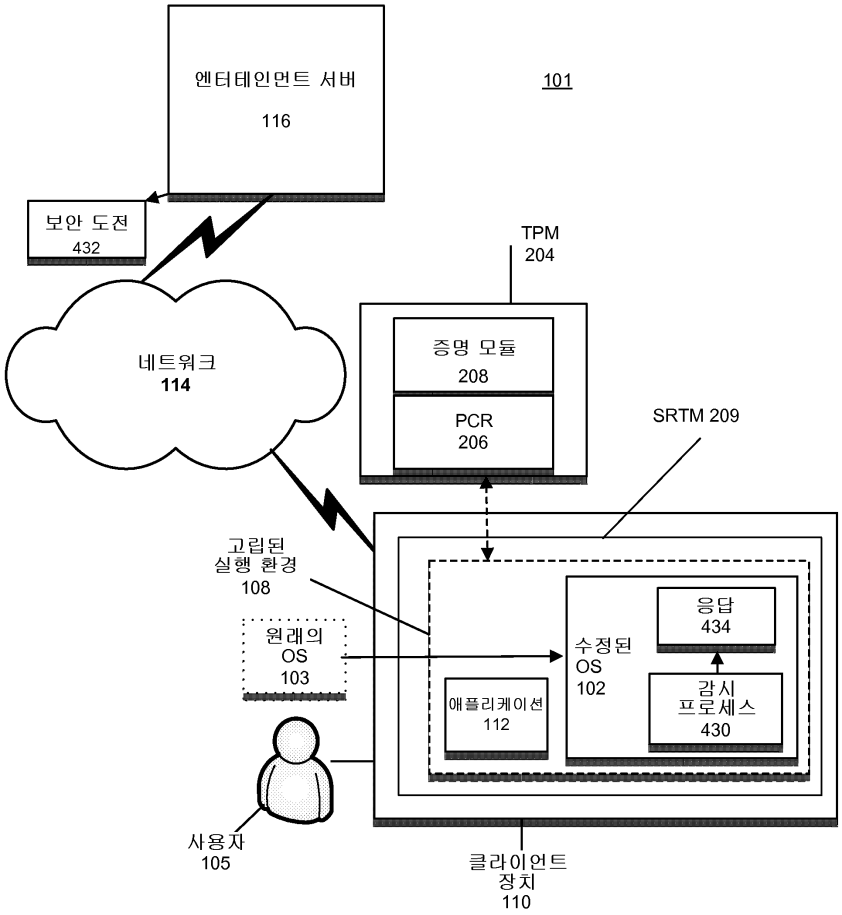
도면3b


322
무결성
보호 모듈
322
324
무결성 보호
디스크 파일
320(1)
무결성 보호
디스크 파일
320(2)
무결성 보호
디스크 파일
320(N)

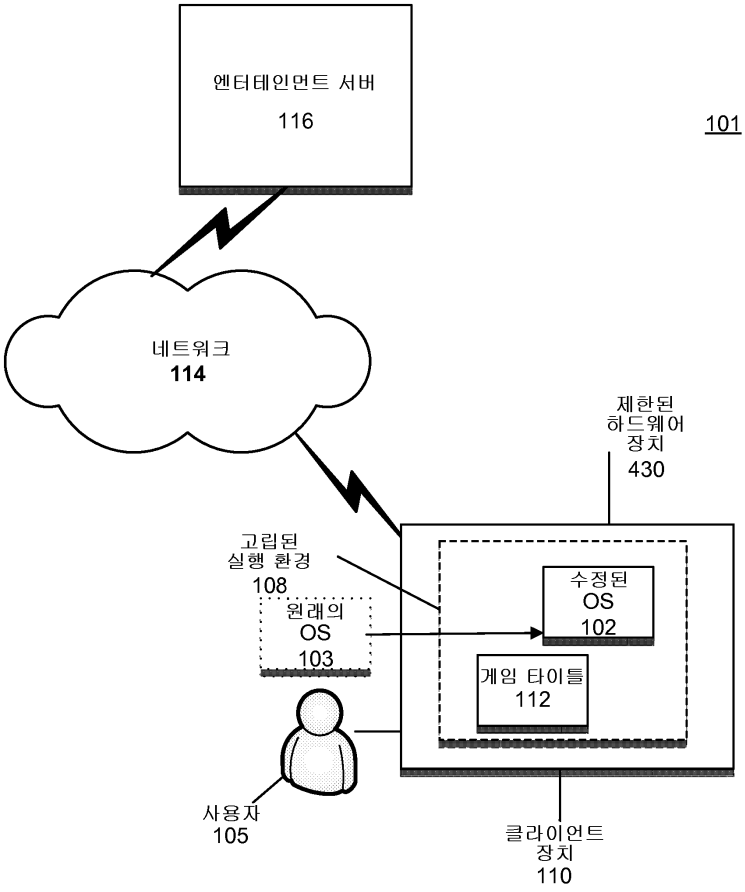
도면4a


라이센스된
보안 채널
410
CPU
402
보안 프로세서
404
코드
406

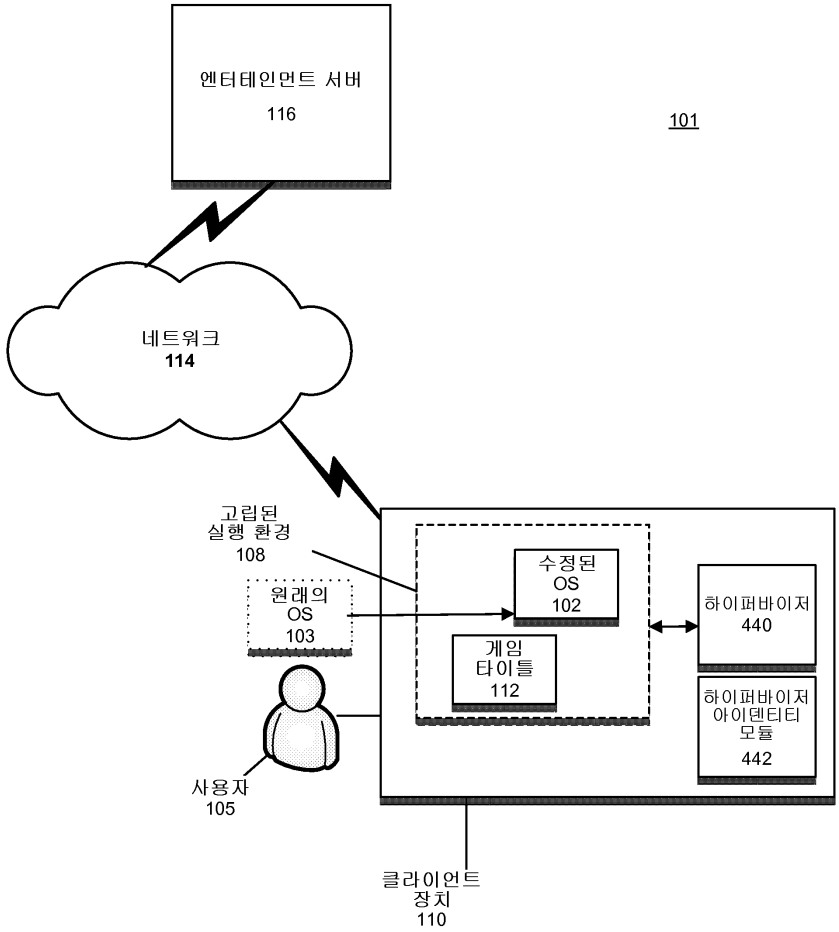
도면4b


엔터테인먼트 서버
116
101
보안 도전
432
TPM
204
네트워크
114
증명 모듈
208
PCR
206
SRTM 209
고립된
실행 환경
108
원래의
OS
103
수정된
OS
102
응답
434
애플리케이션
112
감시
프로세스
430
사용자
105
클라이언트
장치
110

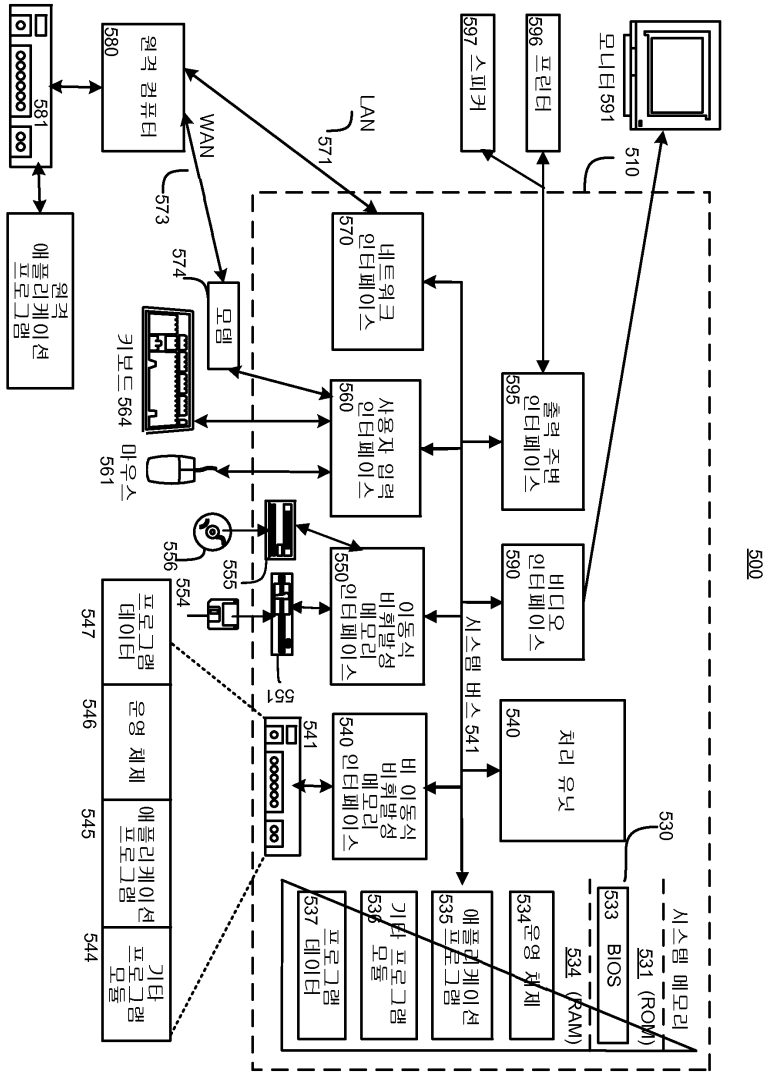
도면4c


엔터테인먼트 서버
116
101
네트워크
114
제한된
하드웨어
장치
430
고립된
실행 환경
108
원래의
OS
103
수정된
OS
102
게임 타이틀
112
사용자
105
클라이언트
장치
110

도면4d


엔터테인먼트 서버
116
101
네트워크
114
고립된
실행 환경
108
원래의
OS
103
수정된
OS
102
게임
타이틀
112
하이퍼바이저
440
하이퍼바이저
아이덴티티
모듈
442
사용자
105
클라이언트
장치
110

도면5


500
510
모니터 591
스피커 597
프린터 596
LAN
571
WAN
573
원격 컴퓨터 580
581
원격 애플리케이션 프로그램
574
모뎀
키보드 564
마우스 561
556
555
554
565
네트워크 인터페이스 570
사용자 입력 인터페이스 560
출력 주변 인터페이스 595
비디오 인터페이스 590
이동식 비휘발성 메모리 인터페이스 550
비 이동식 비휘발성 메모리 인터페이스 540
541
시스템 버스 541
처리 유닛 540
530
시스템 메모리
531 (ROM)
533 BIOS
534 (RAM)
운영 체제 534
애플리케이션 프로그램 535
기타 프로그램 모듈 536
프로그램 데이터 537
547
프로그램 데이터
546
운영 체제
545
애플리케이션 프로그램
544
기타 프로그램 모듈