(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2017-0129794
(43) 공개일자 2017년11월27일

(51) 국제특허분류(Int. Cl.)
G06F 9/455 (2006.01) *G06F 21/53* (2013.01)
G06F 9/45 (2006.01)
(52) CPC특허분류
G06F 9/45516 (2013.01)
G06F 21/53 (2013.01)
(21) 출원번호 10-2017-7027848
(22) 출원일자(국제) 2015년04월10일
심사청구일자 2017년10월26일
(85) 번역문제출일자 2017년09월28일
(86) 국제출원번호 PCT/IB2015/000883
(87) 국제공개번호 WO 2016/162720
국제공개일자 2016년10월13일

(71) 출원인
구글 엘엘씨
미국 캘리포니아 마운틴 뷰 엠피시어터 파크웨이 1600 (우:94043)
(72) 발명자
엘츠신, 예브게니
러시아 115035 모스코 발충 스트리트 7
이고티, 니콜라이
러시아 115035 모스코 발충 스트리트 7
(뒷면에 계속)
(74) 대리인
특허법인 남앤드남

전체 청구항 수 : 총 20 항

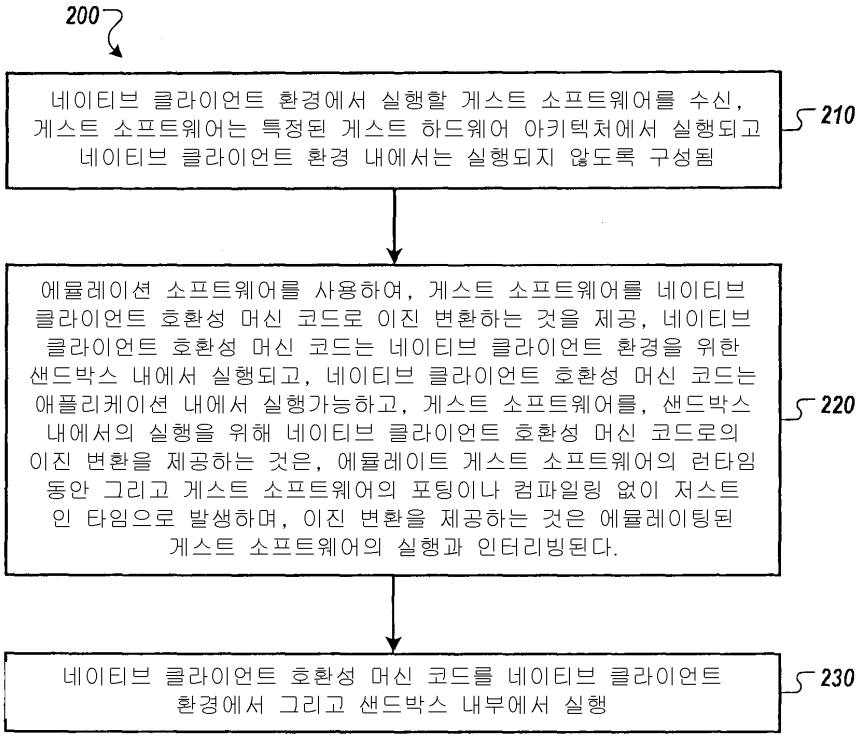
(54) 발명의 명칭 **네이티브 클라이언트로의 이진 변환**

(57) 요 약

이진 변환을 위한 시스템들 및 방법들이 개시된다. 일부 구현들에서, 네이티브 클라이언트 환경에서 실행하는 게스트 소프트웨어가 수신된다. 게스트 소프트웨어는 특정된 게스트 하드웨어 아키텍처에서 실행되고 네이티브 클라이언트 환경 내에서는 실행되지 않도록 구성된다. 게스트 소프트웨어를 네이티브 클라이언트 호환성 머신

(뒷면에 계속)

대 표 도 - 도2



코드로 이진 변환하는 것이 에뮬레이션 소프트웨어를 사용하여 제공된다. 네이티브 클라이언트 호환성 머신 코드는 네이티브 클라이언트 환경을 위한 샌드박스 내에서 실행된다. 네이티브 클라이언트 호환성 머신 코드는 애플리케이션 내에서 실행가능하다. 게스트 소프트웨어를 샌드박스 내에서의 실행을 위해 네이티브 클라이언트 호환성 머신 코드로 이진 변환하는 것을 제공하는 것은, 에뮬레이팅된 게스트 소프트웨어의 런타임 동안, 그리고 게스트 소프트웨어의 포팅 또는 리컴파일링 없이 저스트 인 타임(just in time)으로 발생한다. 이진 변환을 제공하는 것은, 에뮬레이팅된 게스트 소프트웨어의 실행과 인터리빙된다.

(52) CPC특허분류
G06F 8/52 (2013.01)
G06F 2221/033 (2013.01)

(72) 발명자
칼야빈, 안드레이
러시아 115035 모스코 발충 스트리트 7
폴루킨, 드미트리
러시아 115035 모스코 발충 스트리트 7

명 세 서

청구범위

청구항 1

방법으로서,

네이티브 클라이언트 환경(Native Client environment)에서 실행할 게스트 소프트웨어를 수신하는 단계 -상기 게스트 소프트웨어는 특정된 게스트 하드웨어 아키텍처에서 실행되고 네이티브 클라이언트 환경 내에서는 실행되지 않도록 구성됨-; 및

에뮬레이션 소프트웨어를 사용하여, 상기 게스트 소프트웨어를 네이티브 클라이언트 호환성 머신 코드로 이진 변환하는 것을 제공하는 단계를 포함하고,

상기 네이티브 클라이언트 호환성 머신 코드는 상기 네이티브 클라이언트 환경을 위한 샌드박스 내에서 실행되고, 상기 네이티브 클라이언트 호환성 머신 코드는 애플리케이션 내에서 실행가능하고, 상기 게스트 소프트웨어를, 샌드박스 내에서의 실행을 위해 네이티브 클라이언트 호환성 머신 코드로 이진 변환하는 것을 제공하는 것은, 에뮬레이팅된 게스트 소프트웨어의 런타임 동안, 그리고 게스트 소프트웨어의 포팅 또는 리컴파일링 없이 저스트 인 타임(just in time)으로 발생할 수 있고, 이진 변환하는 것을 제공하는 단계는, 상기 에뮬레이팅된 게스트 소프트웨어의 실행과 인터리빙되는, 방법.

청구항 2

제 1 항에 있어서,

상기 네이티브 클라이언트 환경을 위한 샌드박스는, 호스트 머신에서 이용가능한 레지스터들에 또는 메모리에 저장된 에뮬레이팅된 게스트 레지스터들의 세트에 액세스하고, 상기 네이티브 클라이언트 호환성 머신 코드와 연관된 데이터가 네이티브 클라이언트 환경을 위한 샌드박스 내에 저장되고, 상기 에뮬레이팅된 게스트 레지스터들은 상기 특정된 게스트 하드웨어 아키텍처의 레지스터들에 해당하는, 방법.

청구항 3

제 1 항에 있어서,

상기 게스트 소프트웨어를 상기 네이티브 클라이언트 호환성 머신 코드로 이진 변환하는 것을 제공하는 단계는,

상기 특정된 게스트 하드웨어 아키텍처에서 실행될 경우 상기 게스트 소프트웨어에 의해 사용되는 레지스터들을 나타내는 가상 레지스터들의 세트를 생성하는 단계를 포함하고,

상기 가상 레지스터들의 상기 어드레스들은 베이스 포인터 (RBP) 플러스 미리결정된 오프셋에 의해 참조되고, 상기 가상 레지스터들의 세트 내의 각각의 가상 레지스터는, 하나의 명령을 통해, 상기 샌드박스 내에서부터 액세스할 수 있는, 방법.

청구항 4

제 1 항에 있어서,

상기 게스트 소프트웨어를 상기 네이티브 클라이언트 호환성 머신 코드로 이진 변환하는 것을 제공하는 단계는,

상기 특정된 게스트 하드웨어 아키텍처의 특징들 및 상기 게스트 소프트웨어의 API(application programming interface) 호들을 상기 네이티브 클라이언트 내에서 에뮬레이팅하는 단계를 포함하는, 방법.

청구항 5

제 1 항에 있어서,

상기 게스트 소프트웨어를 상기 네이티브 클라이언트 호환성 머신 코드로 이진 변환하는 것을 제공하는 단계는,

상기 게스트 소프트웨어의 코드를, 상기 네이티브 클라이언트 환경과 호환가능한 코드로 대체하는 단계를 포함

하는, 방법.

청구항 6

제 1 항에 있어서,

상기 게스트 소프트웨어는 ARM 소프트웨어 또는 x86 소프트웨어를 포함하는, 방법.

청구항 7

제 1 항에 있어서,

상기 네이티브 클라이언트 호환성 머신 코드는 안전하고 포팅가능하며, 상기 게스트 소프트웨어는 안전하지 않거나 포팅가능하지 않은, 방법.

청구항 8

제 1 항에 있어서,

상기 게스트 소프트웨어는 특정 게스트 운영 시스템 내에서만 실행하도록 구성되며, 상기 네이티브 클라이언트 환경은 다수의 상이한 운영 시스템들 중 임의의 운영 시스템 내에서 실행되도록 구성되는, 방법.

청구항 9

제 1 항에 있어서,

상기 애플리케이션은 브라우저인, 방법.

청구항 10

명령들을 포함하는 비일시적 컴퓨터 판독가능 매체로서,

상기 명령들은, 하나 또는 그 초과의 컴퓨터들에 의해 실행될 경우, 상기 하나 또는 그 초과의 컴퓨터들로 하여금 방법들을 구현하게 하며,

상기 방법은,

네이티브 클라이언트 환경에서 실행할 게스트 소프트웨어를 수신하는 단계 -상기 게스트 소프트웨어는 특정된 게스트 하드웨어 아키텍처에서 실행되고 네이티브 클라이언트 환경 내에서는 실행되지 않도록 구성됨-; 및

에뮬레이션 소프트웨어를 사용하여, 상기 게스트 소프트웨어를 네이티브 클라이언트 호환성 머신 코드로 이진 변환하는 것을 제공하는 단계를 포함하고,

상기 네이티브 클라이언트 호환성 머신 코드는 상기 네이티브 클라이언트 환경을 위한 샌드박스 내에서 실행되고, 상기 네이티브 클라이언트 호환성 머신 코드는 애플리케이션 내에서 실행가능하고, 상기 게스트 소프트웨어를, 샌드박스 내에서의 실행을 위해 네이티브 클라이언트 호환성 머신 코드로 이진 변환하는 것을 제공하는 것은, 에뮬레이팅된 게스트 소프트웨어의 런타임 동안, 그리고 게스트 소프트웨어의 포팅 또는 리컴파일링 없이 게스트 인 타임으로 발생할 수 있고, 이진 변환하는 것을 제공하는 단계는, 상기 에뮬레이팅된 게스트 소프트웨어의 실행과 인터리빙되며,

상기 게스트 소프트웨어를 네이티브 클라이언트 호환성 머신 코드로 이진 변환하는 것을 제공하는 단계는,

상기 특정된 게스트 하드웨어 아키텍처에서 실행될 경우 상기 게스트 소프트웨어에 의해 사용되는 레지스터들을 나타내는 가상 레지스터들의 세트를 생성하는 단계를 포함하고,

상기 가상 레지스터들의 상기 어드레스들은 베이스 포인터 (RBP) 플러스 미리결정된 오프셋에 의해 참조되고, 상기 가상 레지스터들의 세트 내의 각각의 가상 레지스터는, 하나의 명령을 통해, 상기 샌드박스 내에서부터 액세스할 수 있는, 명령들을 포함하는 비일시적 컴퓨터 판독가능 매체.

청구항 11

제 9 항에 있어서,

상기 네이티브 클라이언트 환경을 위한 샌드박스는, 호스트 머신에서 이용가능한 레지스터들에 또는 메모리에

저장된 에뮬레이팅된 게스트 레지스터들의 세트에 액세스하고,

상기 네이티브 클라이언트 호환성 머신 코드와 연관되는 데이터는 상기 네이티브 클라이언트 환경을 위한 샌드박스 내에 저장되고, 상기 에뮬레이팅된 게스트 레지스터들은 상기 특정된 게스트 하드웨어 아키텍처의 레지스터들에 해당하는, 명령들을 포함하는 비일시적 컴퓨터 판독가능 매체.

청구항 12

제 9 항에 있어서,

상기 게스트 소프트웨어를 상기 네이티브 클라이언트 호환성 머신 코드로 이진 변환하는 것을 제공하는 단계는,

상기 특정된 게스트 하드웨어 아키텍처의 특징들 및 상기 게스트 소프트웨어의 API(application programming interface) 호들을 상기 네이티브 클라이언트 내에서 에뮬레이팅하는 단계를 포함하는, 명령들을 포함하는 비일시적 컴퓨터 판독가능 매체.

청구항 13

제 9 항에 있어서,

상기 게스트 소프트웨어를 상기 네이티브 클라이언트 호환성 머신 코드로 이진 변환하는 것을 제공하는 단계는,

상기 게스트 소프트웨어의 코드를, 상기 네이티브 클라이언트 환경과 호환가능한 코드로 대체하는 단계를 포함하는, 명령들을 포함하는 비일시적 컴퓨터 판독가능 매체.

청구항 14

제 9 항에 있어서,

상기 게스트 소프트웨어는 ARM 소프트웨어 또는 x86 소프트웨어를 포함하는, 명령들을 포함하는 비일시적 컴퓨터 판독가능 매체.

청구항 15

제 9 항에 있어서,

상기 네이티브 클라이언트 호환성 머신 코드는 안전하고 포팅가능하며, 상기 게스트 소프트웨어는 안전하지 않거나 포팅가능하지 않은, 명령들을 포함하는 비일시적 컴퓨터 판독가능 매체.

청구항 16

시스템으로서,

하나 또는 그 초과의 프로세서들; 및

명령들을 포함하는 메모리를 포함하고,

상기 명령들은, 상기 하나 또는 그 초과의 프로세서들에 의해 실행될 경우, 상기 하나 또는 그 초과의 프로세서들로 하여금 방법을 구현하게 하며,

상기 방법은,

네이티브 클라이언트 환경에서 실행할 게스트 소프트웨어를 수신하는 단계 -상기 게스트 소프트웨어는 특정된 게스트 하드웨어 아키텍처에서 실행되고 네이티브 클라이언트 환경 내에서는 실행되지 않도록 구성됨-; 및

에뮬레이션 소프트웨어를 사용하여, 상기 게스트 소프트웨어를 네이티브 클라이언트 호환성 머신 코드로 이진 변환하는 것을 제공하는 단계를 포함하고,

상기 네이티브 클라이언트 호환성 머신 코드는 상기 네이티브 클라이언트 환경을 위한 샌드박스 내에서 실행되고, 상기 네이티브 클라이언트 호환성 머신 코드는 애플리케이션 내에서 실행가능하고, 상기 게스트 소프트웨어를 샌드박스 내에서의 실행을 위해 네이티브 클라이언트 호환성 머신 코드로 이진 변환하는 것을 제공하는 것은, 에뮬레이팅된 게스트 소프트웨어의 런타임 동안, 그리고 게스트 소프트웨어의 포팅 또는 리컴파일링 없이 저스트 인 타임으로 발생할 수 있고, 상기 이진 변환하는 것을 제공하는 단계는, 상기 에뮬레이팅된 게스트 소

프트웨어의 실행과 인터리빙되고, 상기 네이티브 클라이언트 환경을 위한 샌드박스는, 호스트 머신에서 이용가능한 레지스터들에 또는 메모리에 저장된 에뮬레이팅된 게스트 레지스터들의 세트에 액세스하고, 상기 네이티브 클라이언트 호환성 머신 코드와 연관되는 데이터는 상기 네이티브 클라이언트 환경을 위한 샌드박스 내에 저장되고, 상기 에뮬레이팅된 게스트 레지스터들은 상기 특정된 게스트 하드웨어 아키텍처의 레지스터들에 해당하는, 시스템.

청구항 17

제 16 항에 있어서,

상기 게스트 소프트웨어를 상기 네이티브 클라이언트 호환성 머신 코드로 이진 변환하는 것을 제공하는 단계는,

상기 특정된 게스트 하드웨어 아키텍처에서 실행될 경우 상기 게스트 소프트웨어에 의해 사용되는 레지스터들을 나타내는 가상 레지스터들의 세트를 생성하는 단계를 포함하고,

상기 가상 레지스터들의 상기 어드레스들은 베이스 포인터 (RBP) 플러스 미리결정된 오프셋에 의해 참조되고, 상기 가상 레지스터들의 세트 내의 각각의 가상 레지스터는, 하나의 명령을 통해, 상기 샌드박스 내에서부터 액세스할 수 있는, 시스템.

청구항 18

제 16 항에 있어서,

상기 게스트 소프트웨어를 상기 네이티브 클라이언트 호환성 머신 코드로 이진 변환하는 것을 제공하는 단계는,

상기 특정된 게스트 하드웨어 아키텍처의 특징들 및 상기 게스트 소프트웨어의 API(application programming interface) 호들을 상기 네이티브 클라이언트 내에서 에뮬레이팅하는 단계를 포함하는, 시스템.

청구항 19

제 16 항에 있어서,

상기 게스트 소프트웨어를 상기 네이티브 클라이언트 호환성 머신 코드로 이진 변환하는 것을 제공하는 단계는,

상기 게스트 소프트웨어의 코드를, 상기 네이티브 클라이언트 환경과 호환가능한 코드로 대체하는 단계를 포함하는, 시스템.

청구항 20

제 16 항에 있어서,

상기 게스트 소프트웨어는 ARM 소프트웨어 또는 x86 소프트웨어를 포함하는, 시스템.

발명의 설명

기 술 분 야

[0001] [0001] 본 출원은, 명칭이 "BINARY TRANSLATION ON SHARED OBJECT LEVEL"이고 대리인 문서 번호 제096553-0073호인, 본원과 동시에 출원된 미국 특허 출원과 관련되며, 상기 출원의 전체 개시내용이 인용에 의해 본원에 포함된다.

배 경 기 술

[0002] [0002] 본 기술은 일반적으로 이진 변환 기술들에 관한 것이다. 일부 소프트웨어는 특정 CPU 아키텍처들, 이를테면, ARM® 또는 x86®과 특정 운영 시스템, 이를 테면, Android® 또는 Microsoft Windows®을 위한 이진 프로그램들로서 컴파일링된다. 이진 프로그램들은 인터넷을 통해 사용자의 컴퓨터로 다운로딩될 수 있다. 그러나, 사용자가 프로그램을 신뢰하지 않을 수 있으며 프로그램을 안전 모드에서 실행되길 원할 수 있는데, 이 프로그램이 프로그램 외부의 컴퓨터에 저장되어 있는 데이터에 대한 액세스를 제한한다. 상기 예시된 바와 같이, 컴퓨터 상에서 소프트웨어를 안전하게 실행시키기 위한 접근법이 바람직할 수 있다.

[0003] [0003] 네이티브 클라이언트(Native Client)는 컴퓨터 상에서 소프트웨어를 안전하게 실행시키는데 사용될 수

있다. 그러나, 일부 소프트웨어는 네이티브 클라이언트 용으로 기록되지 않았으며 네이티브 클라이언트와 호환성이 없다. 이러한 소프트웨어는 리컴파일(recompile)되고 네이티브 클라이언트에 포팅될 필요가 있을 수 있으며, 이는 일부 대형 최신 소프트웨어 제품들의 경우에는 적지 않은 노력을 요구할 수 있다. 상기 예시된 바와 같이, 네이티브 클라이언트와 호환성이 없는 코드를 네이티브 클라이언트로 변환하는 것이 바람직할 수 있다.

발명의 내용

[0004] [0004] 일부 양상에 따르면, 본 기술은 방법과 관련된다. 방법은 네이티브 클라이언트 환경에서 실행할 게스트 소프트웨어를 수신하는 단계를 포함하며, 게스트 소프트웨어는 특정된 게스트 하드웨어 아키텍처에서 실행되고 네이티브 클라이언트 환경 내에서는 실행되지 않도록 구성된다. 방법은 에뮬레이션 소프트웨어를 사용하여, 게스트 소프트웨어를 네이티브 클라이언트 호환성 머신 코드로 이진 변환하는 것을 제공하는 단계를 포함하고, 네이티브 클라이언트 호환성 머신 코드는 네이티브 클라이언트 환경을 위한 샌드박스 내에서 실행되고, 네이티브 클라이언트 호환성 머신 코드는 애플리케이션 내에서 실행가능하고, 게스트 소프트웨어를 샌드박스 내에서의 실행을 위해 네이티브 클라이언트 호환성 머신 코드로 이진 변환하는 것을 제공하는 것은, 에뮬레이팅된 게스트 소프트웨어의 런타임 동안, 그리고 게스트 소프트웨어의 포팅 또는 리컴파일링 없이 저스트 인 타임(just in time)으로 발생되며, 이진 변환을 제공하는 단계는, 에뮬레이팅된 게스트 소프트웨어의 실행과 인터리빙된다. 방법은 네이티브 클라이언트 환경에서 그리고 샌드박스 내에서 네이티브 클라이언트 호환성 머신 코드를 실행하는 단계를 포함한다.

[0005] [0005] 일부 양상에 따르면, 본 기술은 명령들을 저장하는 비일시적 컴퓨터 판독가능 매체와 관련된다. 명령들은 네이티브 클라이언트 환경에서 실행할 게스트 소프트웨어를 수신하기 위한 코드를 포함하며, 게스트 소프트웨어는 특정된 게스트 하드웨어 아키텍처에서 실행되고 네이티브 클라이언트 환경 내에서는 실행되지 않도록 구성된다. 명령들은, 에뮬레이션 소프트웨어를 사용하여, 게스트 소프트웨어를 네이티브 클라이언트 호환성 머신 코드로 이진 변환하는 것을 제공하기 위한 코드를 포함하고, 네이티브 클라이언트 호환성 머신 코드는 네이티브 클라이언트 호환성 머신 코드를 위한 샌드박스 내에서 실행되고, 네이티브 클라이언트 호환성 머신 코드는 애플리케이션 내에서 실행가능하고, 게스트 소프트웨어를 샌드박스 내에서의 실행을 위해 네이티브 클라이언트 호환성 머신 코드로 이진 변환하는 것을 제공하는 것은, 에뮬레이팅된 게스트 소프트웨어의 런타임 동안, 그리고 게스트 소프트웨어의 포팅 또는 리컴파일링 없이 저스트 인 타임으로 발생할 수 있고, 이진 변환을 제공하는 단계는, 에뮬레이팅된 게스트 소프트웨어의 실행과 인터리빙된다. 게스트 소프트웨어의 네이티브 클라이언트 호환성 머신 코드로의 이진 변환을 제공하는 것은, 특정된 게스트 하드웨어 아키텍처에서 실행될 경우 게스트 소프트웨어에 의해 사용되는 레지스터들을 나타내는 가상 레지스터들의 세트를 생성하는 것을 포함하고, 가상 레지스터들의 어드레스들은 베이스 포인터(RBP) 플러스 미리결정된 오프셋에 의해 참조되고, 가상 레지스터들의 세트 내의 각각의 가상 레지스터는, 하나의 명령을 통해, 샌드박스 내에서부터 액세스할 수 있다.

[0006] [0006] 일부 양상에 따르면, 본 기술은 시스템과 관련된다. 시스템은 하나 또는 그 초과의 프로세서들 및 명령들을 저장하는 메모리를 포함한다. 명령들은 네이티브 클라이언트 환경에서 실행할 게스트 소프트웨어를 수신하기 위한 코드를 포함하며, 게스트 소프트웨어는 특정된 게스트 하드웨어 아키텍처에서 실행되고 네이티브 클라이언트 환경 내에서는 실행되지 않도록 구성된다. 명령들은 에뮬레이션 소프트웨어를 사용하여, 게스트 소프트웨어를 네이티브 클라이언트 호환성 머신 코드로 이진 변환하는 것을 제공하기 위한 코드를 포함하고, 네이티브 클라이언트 호환성 머신 코드는 네이티브 클라이언트 환경을 위한 샌드박스 내에서 실행되고, 네이티브 클라이언트 호환성 머신 코드는 애플리케이션 내에서 실행가능하고, 게스트 소프트웨어를 샌드박스 소프트웨어 내에서의 실행을 위해 네이티브 클라이언트로 이진 변환하는 것을 제공하는 것은, 에뮬레이팅된 게스트 소프트웨어의 런타임 동안, 그리고 게스트 소프트웨어의 포팅 또는 리컴파일링 없이 저스트 인 타임으로 발생되며, 네이티브 클라이언트 환경을 위한 샌드박스는, 호스트 시스템에서 이용가능한 레지스터들에 또는 메모리에 저장된 에뮬레이팅된 게스트 레지스터들의 세트에 액세스하고, 네이티브 클라이언트 호환성 머신 코드와 연관되는 데이터는 네이티브 클라이언트 환경을 위한 샌드박스 내에 저장되고, 에뮬레이팅된 게스트 레지스터들은 특정된 게스트 하드웨어 아키텍처의 레지스터들에 해당한다.

[0007] [0007] 본 기술의 다른 구성들이 다음의 상세한 설명으로부터 쉽게 자명해질 것이고, 여기서 본 기술의 다양한 구성들은 예시로서 도시되고 설명된다는 것을 이해한다. 인식되는 바와 같이, 본 기술은 다른 구성 및 상이한 구성이 가능할 수 있고, 그것의 몇 가지 세부사항들이 다양한 다른 관점들에서 수정될 수 있으며, 이들 모두는 본 기술의 범위로부터 벗어나지 않는다. 따라서, 도면들 및 상세한 설명은 본질적으로 예시를 위한 것으로 간주되며 제한적인 것으로 간주되지 않는다.

도면의 간단한 설명

[0008] [0008] 본 기술의 특징들이 첨부된 청구항들에서 제시된다. 그러나, 설명을 위해, 개시되는 청구대상의 몇 가지 양상들이 아래의 도면들에서 제시된다.

[0009] [0009] 도 1a는 네이티브 클라이언트로의 이진 변환과 관련될 수 있는 예시적인 게스트 머신을 예시한다.

[0010] [0010] 도 1b는 네이티브 클라이언트로의 이진 변환과 관련될 수 있는 예시적인 호스트 머신을 예시한다.

[0011] [0011] 도 2는 네이티브 클라이언트로의 이진 변환이 완료될 수 있는 예시적인 프로세스를 예시한다.

[0012] [0012] 도 3은 본 기술의 일부 구현들이 구현되는 예시적인 전자 시스템을 개념적으로 예시한다.

발명을 실시하기 위한 구체적인 내용

[0009] [0013] 아래에 제시되는 상세한 설명은 본 기술의 다양한 구성들에 대한 설명으로서 의도되며, 본 기술이 실시될 수 있는 유일한 구성들을 나타내도록 의도되지 않는다. 첨부된 도면들은 본 명세서에 통합되고 상세한 설명의 일부를 구성한다. 상세한 설명은 본 기술의 완전한 이해를 제공할 목적으로 특정 세부 사항들을 포함한다. 그러나, 본 기술은 본원에 제시된 특정 상세들에 한정되지 않으며 이러한 특정 상세들 없이도 실시될 수 있다는 것이 분명하고 명백할 것이다. 일부 예들에서, 특정 구조들 및 컴포넌트들은 본 기술의 개념들을 모호하게 하는 것을 방지하기 위해 블록도 형태로 도시된다.

[0010] [0014] 본원에서 사용되는 바와 같이, "네이티브 클라이언트"는, 소프트웨어 결함 격리 및 이진 코드 검증 접근법들을 전개하는 샌드박싱 환경들을 지칭할 수 있다. 네이티브 클라이언트는 POSIX(Portable Operating System Interface)와 유사한 API(application programming interface)들의 제한된 세트를 구현하고 웹 브라우저 내에서 실행될 수 있다. "이진 변환"은 하나의 플랫폼(게스트) 상에서 실행되는 애플리케이션 또는 운영 시스템의 머신 코드(게스트 코드로 지칭함)를 분석하여 상이한 플랫폼(호스트) 상에서 실행하기에 적합한 코드를 생성하는 메커니즘을 지칭할 수 있다. "플랫폼"은 하드웨어 및 소프트웨어 스택 조합을 포함할 수 있다. 애플리케이션들은 통상적으로 특정 플랫폼, 이를 테면, ARM 또는 x86 용으로 개발된다. 상기 용어들 각각은 또한, 그의 명료하고 그리고 일반적인 의미를 포함한다.

[0011] [0015] 본 기술은, 게스트 플랫폼(예를 들어, Android ARM) 용으로 개발된 기존 소프트웨어를 다른 플랫폼, 즉, 더 나은 API-레벨 포팅가능성(portability), 안전 및 신뢰 특징들을 갖는 호스트 플랫폼(예를 들어, 네이티브 클라이언트 x86-64)에서 실행하기 위한 이진 변환 기술들과 관련된다. 게스트 플랫폼을 위한 애플리케이션은, 동적으로 로딩된 라이브러리들의 형태로 네이티브 컴포넌트들에 액세스할 수 있는 포팅가능한 프로그래밍 언어(예를 들어, 자바(Java))로 기록될 수 있다. 네이티브 컴포넌트들은 공유 오브젝트들로 지칭될 수 있으며, 게스트 플랫폼의 하드웨어 아키텍처 및 운영 시스템을 사용하여 게스트 플랫폼에서 컴파일링될 수 있다. 애플리케이션 지원하기 위해서, 호스트 플랫폼은 자바 및 네이티브 코드 둘 모두를 모두 실행할 필요가 있을 수 있다. 본 기술은, 네이티브 클라이언트 샌드박스 내부에서, 포팅가능하지 않은 게스트 플랫폼-특정 애플리케이션 라이브러리들을 실행하는 것과 관련된다.

[0012] [0016] 본 기술의 일부 구현들에 따르면, 네이티브 클라이언트 샌드박스 내에서 게스트 코드를 실행시키기 위해서, 네이티브 클라이언트 샌드박싱 규칙들을 준수하는 호스트 코드가 생성된다. 합리적인 성능을 달성하기 위해서, 최적화 기술들이 적용될 수 있다. 최적화 기술들은, 게스트 코드의 거동을 에뮬레이팅하는 저스트-인-타임 SFI(software fault isolation) 준수 호스트 코드를 생성하는 것을 포함할 수 있다. 최적화 기술은, 에뮬레이팅된 게스트 레지스터를 고속 저장소, 이를 테면, 호스트 레지스터들 내 또는 베이스 포인터(RBP) 관련 메모리 내의 캐시에 저장하는 것을 포함할 수 있다. 이러한 호스트 코드의 생성은, 에뮬레이팅된 게스트 컨텍스트에 대한 액세스를 명시적으로 샌드박싱하지 않고도 고속 액세스를 허용한다. 최적화 기술은, 호스트의 공유 오브젝트들을 사용하여 외부 플랫폼 공유 오브젝트들에 대한 호(call)들을 에뮬레이팅하는 것을 포함할 수 있는데, 즉, 가능하다면, 표준 라이브러리들의 상술된 이진 변환을 포함할 수 있다. 이 에뮬레이션 전략은, 명칭이 "BINARY TRANSLATION ON SHARED OBJECT LEVEL"이고 대리인 문서 번호 제093054-0893호인, 본원과 동시에 출원된 미국 특허 출원에 상세히 설명되며, 상기 출원의 전체 개시내용이 인용에 의해 본원에 포함된다.

[0013] [0017] 어떤 경우들에서, 호스트 플랫폼은, 예를 들어, 안전 요건들로 인해, 게스트 플랫폼에서 이용가능한 일부 API들에 액세스하지 못할 수 있다. 이용가능하지 않은 기능성을 에뮬레이팅하기 위한 추가 기술들이 사용된다.

[0014] [0018] 일부 CPU(central processing unit) 아키텍처들은, 페이지 테이블의 적절한 비트(X-비트)를 통해 실행 가능한 코드의 명시적인 마킹을 지원한다. 가상 메모리로부터의 코드는 X-비트가 세팅된 경우에만 실행될 수 있다. 네이티브 클라이언트 샌드박스 내에서부터 X-비트 조작에 대한 직접 지원은, 일부 경우들에서, 안전 상의 이유로 허용되지 않을 수 있다. 또한, 이진 변환기에 의한 게스트 코드 실행은 게스트 코드 상의 호스트 X 비트의 존재를 고려하지 않기 때문에, 그러한 조작들은, 일부 경우들에서, 원하는 효과를 내지 못할 수 있다. X-비트(예를 들어, 통상적으로, POSIX 시스템들 상에서 mmap(PROT_EXEC) API를 사용하여 제공됨)의 조작과 관련되는 기능을 에뮬레이팅하기 위해서, 게스트 mmap() API에 의해 조작된 추가 구조들이 이진 변환 엔진에 의해 고려될 필요가 있다. 이는, 특정 게스트 페이지가 실행가능한지 여부(예를 들어, 특정 게스트 페이지가 그의 X-비트 세트를 갖는지 여부)를 저장하는 메모리에서 비트맵을 통해 달성될 수 있다. 모든 각각의 명령의 실행 시 비트맵의 체크를 방지하기 위해서, 변환 시간 동안 X-비트 체크들이 발생할 수 있다. 이후, 실행가능한 페이지들에 대해 X-비트를 변경하는 것은 코드 캐시의 플러시(flush)를 유발할 수 있다.

[0015] [0019] 네이티브 클라이언트는 신호들의 POSIX 특징을 지원하지 않을 수 있다. 크래시(crash) 또는 실행 실패와 같이, 애플리케이션에서의 예외적인 상황들에 대한 정보를 제공하기 위해 신호들이 사용될 수 있다. 또한, 가비지(garbage) 수집 및 관리되는 언어 런타임들을 위해 신호들이 또한 사용될 수 있다. 신호들을 사용하는 소프트웨어는, 일부 경우들에서, 이진 변환 엔진이 신호 처리기들의 셋팅 및 게스트-특정 신호 처리기들로의 신호들의 전달을 에뮬레이팅하지 않는 경우, 네이티브 클라이언트 샌드박스에서 실행되지 못할 수 있다. 따라서, 신호들을 에뮬레이팅하기 위해서, 이진 변환 엔진은 활성 신호 마스크를 유지할 수 있다. 신호가 전달될 경우, 네이티브 클라이언트 내의 이진 변환 소프트웨어의 실행이 일시중단되고, 신호 처리기가 실행될 수 있다. 신호 처리기가 실행된 후, 이진 변환 소프트웨어의 실행이 재개될 수 있다. 신호들의 이러한 프로세싱은, 이진 변환 디스패처 내부의 신호 마스크의 체크들을 추가하고, 함수 랩핑 메커니즘을 사용하여 적절한 아규먼트(argument)들에 의해 신호 처리기를 명시적으로 호출함으로써 달성될 수 있다.

[0016] [0020] 네이티브 클라이언트는, 네이티브 컴파일된 코드(예를 들어, 일부 경우들에서, 컴파일된 C 또는 C ++ 코드)를 브라우저에서 안전하게 실행시키기 위한 샌드박싱 기술이다. 본원에서 사용되는 바와 같이, 문구들 "샌드박스" 또는 "샌드박싱 기술"은, 프로그램이 액세스할 수 있는 메모리 내 공간을 제한함으로써 실행 중인 프로그램들을 분리하기 위한 컴퓨터 안전 메커니즘을 지칭할 수 있다. 샌드박스 프로그램은, "샌드박스 내부의" 메모리 영역에는 액세스할 수 있고, "샌드박스 외부의" 메모리 영역에는 액세스할 수 없다고 한다. 네이티브 클라이언트 코드는 운영 시스템과는 독립적이며, 운영 시스템에 대한 지원이 구현되는 한 어떤 운영 시스템에서도 실행될 수 있다. 네이티브 클라이언트는, 프로그래머가, 브라우저의 안전 특징들을 유지하면서 웹 브라우저를 통해 임의의 클라이언트 시스템의 네이티브 코드를 개발, 배포 및 실행하도록 허용한다. 네이티브 클라이언트는 네이티브 코드를 샌드박스 내에서 실행할 능력을 기능을 제공하며, 브라우저에 대한 또는 샌드박스 외부의 호스트 운영 시스템에 대한 직접 액세스를 허용하지 않는다. 네이티브 클라이언트는, 제어된 API들을 통해 호스트 운영 시스템에 또는 브라우저에, 제한된 액세스를 제공할 수 있다. 네이티브 클라이언트 호환성 머신 코드로의 이진 변환은, 합법적인 샌드박스 액세스만 발생할 것을 보장한다. 본 기술의 양상들은 브라우저와 관련하여 본원에 설명되었지만, 대안적인 구현들에서, 다른 애플리케이션이 브라우저 대신 사용될 수 있다.

[0017] [0021] 본 기술은, 네이티브 클라이언트와 호환되지 않는 소프트웨어를 네이티브 클라이언트 호환성 머신 코드로 변환한다. 유리하게도, 사용자는, 그/그녀가 신뢰할 수 없는 소스로부터 소프트웨어를 실행할 수 있고 그리고 그 소프트웨어가 샌드박스 내에서 실행될 것이고 샌드박스 외부의 사용자의 컴퓨터에 저장된 정보에 대한 액세스는 제한될 것이라는 것을 보장받을 수 있다. 변환이, 네이티브 클라이언트 내부에서 실행되는 에뮬레이팅된 게스트 소프트웨어의 런타임 동안 발생한다. 코드가 실행되는 때에, "저스트 인 타임" 변환이 코드의 런타임 동안 발생한다. 결과적으로, 실행되고 변환될 필요가 있는 코드만 변환된다. 실행되지 않은 코드의 부분들은 변환되지 않음으로써, 시간과 프로세싱 리소스들이 절약된다. 이진 변환 기술이 중요한 포팅 계층과 함께 전개될 수 있다. 이진 변환기는 게스트 소프트웨어의 코드를 변환할 수 있다. 그러나, 표준 라이브러리 호들은, 에뮬레이션 소프트웨어에서 제공된 포팅 계층을 사용하여 실행될 수 있다. 포팅 계층은, 게스트 플랫폼의 예상되는 API를 네이티브 클라이언트 상의 이용가능한 API들의 세트로 변환할 수 있다. 일부 경우들에서, 이러한 변환은, 동기식 차단 라이브러리 호들을 네이티브 클라이언트의 비동기 비-차단 호들로 변환하는 것뿐만 아니라 네이티브 클리이언트 내에서 네이티브 클리이언트에 누락되어 있는 게스트 하드웨어의 특징들을 에뮬레이팅하는 것을 포함할 수 있다.

[0018] [0022] 이진 변환은, 추가 검증 및 샌드박싱을 위해 적절한 포맷으로 게스트 소프트웨어를 변환할 수 있다. 더 나은 포팅가능성과 안전을 달성하기 위해 SFI(Software fault isolation) 기술이 사용될 수 있다.

[0019] [0023] 본 기술은 이진 변환의 특수한 경우로 간주될 수 있으며, 네이티브 클라이언트는 호스트의 운영 시스템이며, 그 결과, 브라우저에서 실행될 수 있는 안전 프로그램들이 어떠한 운영 시스템 내에서도 실행되게 한다. 네이티브 클라이언트로의 변환의 결과로서, 소프트웨어의 안전 및 포팅가능성 둘 모두가 증가된다.

[0020] [0024] 이진 변환 동안, 코드는 네이티브 클라이언트의 규칙들과 호환가능한 포맷으로 변환될 수 있다. 예를 들어, 네이티브 클라이언트는, 32의 배수인 어드레스들과 교차하는 명령들이 제공되지 않을 것을 요구한다(즉, 명령은 번들 정렬된다). 결과적으로, 이진 변환기가 머신 코드를 방출할 경우, 이진 변환기는, 번들 경계(bundle boundary)(32의 배수)가 결코 교차되지 않을 것을 보장한다.

[0021] [0025] 본 기술의 일부 양상들에서, 컴퓨터는 네이티브 클라이언트 환경에서 실행할 게스트 소프트웨어를 수신한다. 게스트 소프트웨어는, 특정된 게스트 하드웨어 아키텍처(예를 들어, x86 하드웨어 또는 ARM 하드웨어)에서 실행되도록 구성된다. 게스트 소프트웨어는 안전하지 않거나 또는 포팅가능하지 않을 수 있다. 컴퓨터 이진법은 게스트 소프트웨어를 네이티브 클라이언트 호환 머신 코드로 변환한다. 네이티브 클라이언트 호환 머신 코드는, 호스트 머신에 저장된 데이터에 대한 네이티브 클라이언트 호환성 머신 코드의 액세스를 제한하는 샌드박스 내에서 실행된다. 네이티브 클라이언트 호환성 머신 코드는, 브라우저를 지원하는 임의의 운영 시스템의 브라우저 내에서 안전하고, 포팅가능하며, 실행가능하다.

[0022] [0026] 샌드박스는 호스트 머신에서 이용가능한 레지스터들의 세트의 서브세트에 액세스할 수 있다. 게스트 레지스터들과 같은 중요한 게스트 데이터가, 레지스터들의 세트의 서브세트를 통해 또는 그 서브세트의 레지스터들로부터의 오프셋에서의 메모리 포지션들을 통해 또는 게스트 소프트웨어로부터 네이티브 클라이언트 호환성 머신 코드로 제공될 수 있다. 이진 변환 코드가 게스트 레지스터들의 집중적인 사용을 요구할 수 있으므로, 이 접근법은, 네이티브 클라이언트 샌드박스의 요건들을 여전히 충족시키면서, 게스트 하드웨어의 고 성능 에뮬레이션을 허용할 수 있다.

[0023] [0027] 게스트 소프트웨어의 네이티브 클라이언트 호환성 머신 코드로의 이진 변환을 제공하는 것은, 호스트 하드웨어 아키텍처에서 실행될 경우 게스트 소프트웨어에 의해 사용되는 레지스터들을 나타내는 가상 레지스터들의 세트를 생성하는 것을 포함할 수 있다. 가상 레지스터들의 어드레스들은 RBP 플러스 미리결정된 오프셋에 의해 참조될 수 있다. 예를 들어, 레지스터 R0은 포지션 RBP에 있을 수 있고, 레지스터 R1은 포지션 RBP + 4 오프셋 바이트에 있을 수 있고, 레지스터 R2는 포지션 RBP + 8 오프셋 바이트에 있을 수 있는 식이다. 결과적으로, 가상 레지스터들의 세트 내의 각각의 가상 레지스터는, 하나의 명령을 통해, 샌드박스 내에서부터 액세스 가능할 수 있다. 예를 들어, 기록 명령이 포맷: WRITE (POSITION, VALUE)를 갖는 경우, 값 "0"을 레지스터 R2에 기록하는 명령은 : WRITE (RBP + 8, 0)로 기록될 수 있다. 이 기술이 생성된 코드의 최대 성능을 발생시키기 때문에 하나의 명령이 유리할 수 있다. 네이티브 클라이언트 샌드박싱 메커니즘들은 RBP-관련 어드레싱에 대한 특정 예외들을 허용한다. 구체적으로, 하나의 [RBP + 4 * N] 명령이 레지스터 번호 N에 액세스하기 위해 이진 변환 코드로 사용될 수 있다. RBP 레지스터가 기본으로 사용된다. 어떤 경우들에서, 메모리 액세스가 샌드박스 내에 있음을 보장하기 위해서 추가 샌드박싱 명령이 요구될 수 있다.

[0024] [0028] 게스트 소프트웨어의 네이티브 클라이언트 호환성 머신 코드로의 이진 변환을 제공하는 것은 특정된 게스트 하드웨어 아키텍처의 특징들 및 게스트 소프트웨어의 API(application programming interface) 호들을 네이티브 클라이언트 내에서 에뮬레이팅하는 것을 포함할 수 있다. 특정된 게스트 하드웨어 아키텍처의 피처들은 레지스터들, 스택들, 포인터들 등을 포함할 수 있다. API 호들은, 시스템 API들 또는 게스트 소프트웨어와 연관된 API들에 대한 호들을 포함할 수 있다. 레지스터들, 스택들, 포인터들 등이 에뮬레이팅되는데, 이는, 네이티브 클라이언트 호환성 머신 코드가, 하드웨어 아이템들과 유사하게 거동하지만 (연관된 하드웨어 컴포넌트들 없이도) 프로세서 및 메모리에서 실행되는 소프트웨어에 존재하는 하드웨어 아이템들의 소프트웨어 표현들을 생성한다는 것을 의미한다. 에뮬레이팅된 컴포넌트들에 액세스하는 호들은, 게스트 머신 상의 물리적 컴포넌트들에 액세스하는 호들과 정확히 동일하거나 또는 유사할 수 있으므로, 수정되지 않은 게스트 코드가 에뮬레이팅된 환경에서 실행되도록 허용된다.

[0025] [0029] 실행가능 비트 (X 비트) 및 비동기 신호 프로세싱과 같은 네이티브 클라이언트 환경 내부에서 이용 가능하지 않은 게스트 아키텍처의 특정 양상을 에뮬레이팅하기 위해, 추가 소프트웨어 에뮬레이션 로직이 사용될 수 있다. 예를 들어, 네이티브 클라이언트는, mmap (PROT_EXEC) 인터페이스를 통해 POSIX 시스템들에 대해 이용가능한 또는 실행가능한 코드 영역들에 걸친 자유로운 제어(liberal control)를 허용하지 않는다. 따라서, 에뮬레이션 소프트웨어는 추가 메커니즘을 통해, 실행가능한 비트들에 대한 제어를 구현할 수 있다. 에뮬레이션 소프트웨어는 별개의 데이터 구조들을 통해 실행가능한 게스트 코드를 파악할 수 있다. 유사하게, 네이티브 클라

이언트는 신호 전달을 지원하지 않으므로, 에뮬레이션 소프트웨어는, 보류 중인 신호들 마스크를 주기적으로 체크하고 신호가 발생하면 신호 처리기를 명시적으로 호출함으로써 비동기 신호들의 전송을 지원할 수 있다. 본원에 사용된 바와 같이, 문구 "에뮬레이션 소프트웨어" 또는 "에뮬레이터"는, 게스트 소프트웨어가 네이티브 클라이언트 환경 내부에서 실행되도록 허용하는 소프트웨어를 지칭한다.

[0026] [0030] 도 1a는 네이티브 클라이언트로의 이진 변환과 관련될 수 있는 예시적인 게스트 머신(100A)을 예시한다. 게스트 머신(100A)은 임의의 컴퓨팅 디바이스, 예를 들어, 랩탑 컴퓨터, 데스크탑 컴퓨터, 태블릿 컴퓨터, 모바일 전화, PDA(personal digital assistant), 전자 음악 플레이어, 스마트 와치, 하나 또는 그 초과의 프로세서들 및 메모리와 결합된 텔레비전 등일 수 있다. 일부 예들에서, 게스트 머신(100A)은 ARM 하드웨어를 갖는다. 대안으로, 게스트 머신(100A)은 x86 하드웨어를 구비할 수 있다.

[0027] [0031] 도시된 바와 같이, 게스트 머신(100A)은 프로세싱 유닛(102A), 네트워크 인터페이스(104A), 및 메모리(106A)를 포함한다. 프로세싱 유닛(102A)은 하나 또는 그 초과의 프로세서들을 포함한다. 프로세싱 유닛(102A)은 CPU(central processing unit), GPU(graphics processing unit), 또는 임의의 다른 프로세싱 유닛으 포함할 수 있다. 프로세싱 유닛(102A)은, 컴퓨터-판독가능 매체, 예를 들어, 메모리(106A)에 저장되는 컴퓨터 명령들을 실행한다. 네트워크 인터페이스(104A)는, 게스트 머신(100A)이 네트워크, 예를 들어, 인터넷, 인트라넷, 셀룰러 네트워크, 근거리 통신망, 광역 통신망, 유선 네트워크, 무선 네트워크, VPN(virtual private network) 등에서 데이터를 송신하고 수신하도록 허용한다. 메모리(106A)는 데이터 및/또는 명령들을 저장한다. 메모리(106A)는 캐시 유닛, 저장소 유닛, 내부 메모리 유닛, 또는 외부 메모리 유닛 중 하나 또는 그 초과의 것일 수 있다. 도시된 바와 같이, 메모리(106A)는 게스트 레지스터들(108A), 게스트 소프트웨어 프로그램(110A), 및 게스트 API들(112A)을 포함한다.

[0028] [0032] 게스트 레지스터들(108A)은, 게스트 머신(100A)의 하드웨어 아키텍처(예를 들어, ARM 또는 x86)와 연관되는 게스트 머신(100A) 상의 레지스터들이다. 레지스터들(108A) 이외에, 메모리(106A)는 다른 하드웨어 아키텍처들, 이를 테면, 스택(들) 또는 포인터(들)를 포함할 수 있다. 게스트 API들(112A)은 게스트 머신(100A) 상에 존재하는 API들이며, API들을 이용하여, 게스트 머신(100A)을 위한 소프트웨어 및 그의 연관된 하드웨어 아키텍처가 기록될 수 있다. 게스트 API들(112A)은, 게스트 머신(100A)의 하드웨어와 연관된 시스템 API들 또는 게스트 소프트웨어 프로그램(110A)과 함께 게스트 소프트웨어 프로그램(110A)의 벤더에 의해 제공된 벤더 API들을 포함할 수 있다.

[0029] [0033] 게스트 소프트웨어 프로그램(110A)은 게스트 머신(100A)의 하드웨어 아키텍처 상에서 실행하고 그리고 게스트 레지스터들(108A)과 그리고 게스트 API들(112A)과 인터페이싱하도록 구성되는 소프트웨어 프로그램이다. 게스트 소프트웨어 프로그램(110A)은, 네이티브 클라이언트 환경에서 또는 게스트 머신 (100A)의 하드웨어 아키텍처와는 상이한 하드웨어 아키텍처 상에서 실행이 불가능할 수 있다. 예를 들어, 게스트 머신(100A)이 ARM 하드웨어를 갖는 경우, 게스트 소프트웨어 프로그램(110A)은 ARM 하드웨어 상에서 실행하지만 x86 하드웨어 상에서는 실행하지 않도록 구성될 수 있다.

[0030] [0034] 도 1b는 공유 오브젝트 레벨 상의 이진 변환과 관련될 수 있는 예시적인 호스트 머신(100B)을 예시한다. 호스트 머신(100B)은 임의의 컴퓨팅 디바이스, 예를 들어, 랩탑 컴퓨터, 데스크탑 컴퓨터, 태블릿 컴퓨터, 모바일 전화, PDA(personal digital assistant), 전자 음악 플레이어, 스마트 와치, 하나 또는 그 초과의 프로세서들 및 메모리와 결합된 텔레비전 등일 수 있다. 호스트 머신(100B)은 게스트 머신(100A)의 하드웨어 아키텍처와는 상이한 하드웨어 아키텍처를 갖는다. 예를 들어, 게스트 머신(100A)의 하드웨어 아키텍처가 ARM 하드웨어인 경우, 호스트 머신(100B)의 하드웨어 아키텍처는 ARM 하드웨어가 아니며, 예를 들어 x86 하드웨어일 수 있다. 대안으로, 호스트 머신(100B)은 게스트 머신(100A)과 동일한 하드웨어 아키텍처를 가질 수 있다.

[0031] [0035] 도시된 바와 같이, 호스트 머신(100B)은 프로세싱 유닛(102B), 네트워크 인터페이스(104B), 및 메모리(106B)를 포함한다. 프로세싱 유닛(102B)은 하나 또는 그 초과의 프로세서들을 포함한다. 프로세싱 유닛(102B)은 CPU(central processing unit), GPU(graphics processing unit), 또는 임의의 다른 프로세싱 유닛을 포함할 수 있다. 프로세싱 유닛(102B)은, 컴퓨터-판독가능 매체, 예를 들어, 메모리(106B)에 저장되는 컴퓨터 명령들을 실행한다. 네트워크 인터페이스(104B)는, 호스트 머신(100B)이 네트워크, 예를 들어, 인터넷, 인트라넷, 셀룰러 네트워크, 근거리 통신망, 광역 통신망, 유선 네트워크, 무선 네트워크, VPN(virtual private network) 등에서 데이터를 송신하고 수신하도록 허용한다. 메모리(106B)는 데이터 및/또는 명령들을 저장한다. 메모리(106B)는 캐시 유닛, 저장소 유닛, 내부 메모리 유닛, 또는 외부 메모리 유닛 중 하나 또는 그 초과의 것일 수 있다. 예시된 바와 같이, 메모리(106B)는 안전 호스트 레지스터(114B), 안전 호스트 콘텐츠(116B), 및

네이티브 클라이언트 샌드박스(120B)를 포함한다.

[0032] [0036] 안전 호스트 레지스터들(114B)은, 게스트 머신(100B)의 하드웨어 아키텍처(예를 들어, ARM 또는 x86)와 연관되는 호스트 시스템(100B)상의 레지스터들이다. 안전 호스트 레지스터들(114B) 이외에, 메모리(106B)는 다른 하드웨어 아키텍처들, 이를 테면, 스택(들) 또는 포인터(들)를 포함할 수 있다. 안전 호스트 콘텐츠(116B)는 호스트 머신(100B)에 저장된 콘텐츠, 이를 테면, 워드 프로세싱 문서들, 사진들, 비디오들, 오디오 파일들 등을 포함한다. 안전 호스트 레지스터(114B) 및 안전 호스트 콘텐츠(116B)는 네이티브 클라이언트 샌드박스(120B) 외부에 있고 네이티브 클라이언트 샌드박스(120B)로부터 액세스가능하지 않다.

[0033] [0037] 네이티브 클라이언트 샌드박스(120B)는, 네이티브 클라이언트 코드가 실행될 수 있는 메모리(106B)의 안전 영역이다. 네이티브 클라이언트 코드는 네이티브 클라이언트 샌드박스(120B) 내에서 실행될 수 있고, 네이티브 클라이언트 샌드박스(120B) 외부의 레지스터들 또는 데이터, 이를 테면, 안전 호스트 레지스터들(114B) 또는 안전 호스트 콘텐츠(116B)에 액세스할 수 없다. 이러한 방식으로, 호스트 머신(100B)의 사용자는 잠재적으로 신뢰할 수 없는 네이티브 클라이언트 코드를 실행할 수 있는 반면, 그의/그녀의 개인 데이터 및 호스트 머신(100B)의 안전 레지스터들이 잠재적으로 신뢰할 수 없는 코드에 액세스가능하지 않아 안전하다.

[0034] [0038] 도시된 바와 같이, 네이티브 클라이언트 샌드박스(120B)는 액세스가능한 호스트 레지스터들(122B), 에뮬레이팅된 게스트 레지스터들(108B), 네이티브 클라이언트 호환성 머신 코드(110B), 및 에뮬레이팅된 게스트 API들(112B)을 포함한다. 네이티브 클라이언트 호환성 머신 코드(110B)는 네이티브 클라이언트로 이진 변환된 게스트 소프트웨어 프로그램(110A)에 대응한다. 네이티브 클라이언트 호환성 머신 코드(110B)는 네이티브 클라이언트 샌드박스(120B) 내에서 실행된다. 다른 네이티브 클라이언트 코드와 유사하게, 네이티브 클라이언트 호환성 머신 코드(110B)는 네이티브 클라이언트 샌드박스(120B) 외부의 레지스터들 또는 데이터, 이를 테면, 안전 호스트 레지스터들(114B) 또는 안전 호스트 콘텐츠(116B)에 액세스할 수 없다. 네이티브 클라이언트 호환성 머신 코드(110B)는 브라우저 내에서, 실질적으로 임의의 하드웨어 또는 운영 시스템 상에서 실행가능하다. 특히, 네이티브 클라이언트 호환성 머신 코드(110B)는 호스트 머신(100B) 상에서 실행될 수 있으며, 호스트 머신(100B)은 게스트 머신(100A)의 하드웨어 아키텍처와는 상이한 하드웨어 아키텍처를 가질 수 있다. 몇몇 경우들에, 게스트 소프트웨어 프로그램(110A)을 네이티브 클라이언트 호환성 머신 코드(110B)로 이진 변환하는 것은, 네이티브 클라이언트 호환성 머신 코드 프로그램의 런타임 동안, 그리고 네이티브 클라이언트 호환성 머신 코드의 포팅 또는 리컴파일링 없이도 저스트 인 타임으로 발생할 수 있다. 결과적으로, 실행되고 변환될 필요가 있는 코드만 변환된다. 실행되지 않은 코드의 부분들은 변환되지 않음으로써, 시간과 프로세싱 리소스들이 절약된다.

[0035] [0039] 에뮬레이팅된 게스트 레지스터들(108B)은, 네이티브 클라이언트 샌드박스(120B) 내에서 실행되는 소프트웨어로 에뮬레이팅된 게스트 머신(100A)의 게스트 레지스터들(108A)에 대응한다. 게스트 머신(100A) 상에서 실행되는 게스트 소프트웨어 프로그램(110A)에 의해 사용되는 스택(들) 또는 포인터(들)와 같은 다른 하드웨어는 또한 네이티브 클라이언트 샌드박스(120B)에서 에뮬레이팅될 수 있다. 레지스터들(108B)이 에뮬레이팅되는데, 이는, 네이티브 클라이언트 호환성 머신 코드가, 하드웨어 아이템들과 유사하게 거동하지만 (연관된 하드웨어 컴포넌트들 없이도) 프로세서 및 메모리에서 실행되는 소프트웨어에 존재하는 하드웨어 아이템들의 소프트웨어 표현들을 생성한다는 것을 의미한다. 에뮬레이팅된 컴포넌트들에 액세스하는 호들은, 게스트 머신 상의 물리적 컴포넌트들에 액세스하는 호들과 정확히 동일하거나 또는 유사할 수 있으므로, 코드가 게스트 머신(100A)으로부터 네이티브 클라이언트 샌드박스(120B)로 용이하게 포팅되도록 허용한다. 일부 경우들에 따르면, 게스트 소프트웨어 프로그램(110A)의 코드는 네이티브 클라이언트 환경과 호환가능한 코드로 대체된다.

[0036] [0040] 에뮬레이팅된 게스트 API들(112B)은, 네이티브 클라이언트 샌드박스(120B) 내에서 실행되는 소프트웨어에서 에뮬레이팅되거나 또는 이진 변환되는, 게스트 머신(100A)의 게스트 API들(112A)에 대응한다. 에뮬레이팅된 게스트 API들(112B)은, 게스트 머신(100A)의 하드웨어와 연관된 시스템 API들 또는 게스트 소프트웨어 프로그램(110A)과 함께 게스트 소프트웨어 프로그램(110A)의 벤더에 의해 제공된 벤더 API들의 에뮬레이션들을 포함할 수 있다. 게스트 소프트웨어 프로그램(110A)이 게스트 머신(100A)에서 게스트 레지스터들(108A) 및 게스트 API들(112A)에 액세스하는 방법과 유사하게, 대응하는 네이티브 클라이언트 호환성 머신 코드(110B)는 네이티브 클라이언트 샌드박스(120B)에서, 에뮬레이팅된 게스트 레지스터들(108B) 및 에뮬레이팅된 게스트 API들(112B)에 액세스할 수 있다.

[0037] [0041] 네이티브 클라이언트 샌드박스(120B)는 또한 액세스가능한 호스트 레지스터들(122B)을 포함한다. 액세스가능한 호스트 레지스터들(122B)은, 네이티브 클라이언트 샌드박스(120B) 내에서부터, 네이티브 클라이언트

샌드박스(120B) 내에서 실행되는 네이티브 클라이언트 호환성 머신 코드(110B)를 포함하는 소프트웨어에 액세스할 수 있는 호스트 머신(100B)의 레지스터들이다. 호스트 머신의 레지스터들은, 네이티브 클라이언트 호환성 머신 코드에 액세스할 수 없는 안전 호스트 레지스터들(114B), 또는 네이티브 클라이언트 호환성 머신 코드에 의해 액세스가능할 수 있는 액세스가능한 호스트 레지스터들(122B)일 수 있다.

[0038] [0042] 도 2는 네이티브 클라이언트로의 이진 변환이 완료될 수 있는 예시적인 프로세스(200)를 예시한다.

[0039] [0043] 프로세스(200)는 단계(210)에서 시작하며, 단계(210)에서, 호스트 머신(예를 들어, 호스트 머신(100B))이 네이티브 클라이언트 환경(예를 들어, 네이티브 클라이언트 샌드박스(120B))에서 실행할 게스트 소프트웨어(예를 들어, 게스트 소프트웨어 프로그램(110A))를 수신한다. 게스트 소프트웨어는 특정된 게스트 하드웨어 아키텍처(예를 들어, 게스트 머신(100A))에서 실행되고 네이티브 클라이언트 환경 내에서는 실행되지 않도록 구성된다. 네이티브 클라이언트 환경은 다수의 상이한 하드웨어 아키텍처들 중 임의의 것에서 실행되도록 구성된다. 예를 들어, 네이티브 클라이언트 환경은, 다수의 상이한 하드웨어 아키텍처들 중 임의의 것에 존재할 수 있는 브라우저와 같은 애플리케이션 내에서 실행되도록 구성될 수 있다.

[0040] [0044] 단계(220)에서, 호스트 머신은, 에뮬레이션 소프트웨어를 사용하여, 게스트 소프트웨어를 네이티브 클라이언트 호환성 머신 코드(예를 들어, 네이티브 클라이언트 호환성 머신 코드(110B))로 이진 변환하는 것을 제공한다. 네이티브 클라이언트 호환성 머신 코드는 네이티브 클라이언트 환경을 위한 샌드박스(예를 들어, 네이티브 클라이언트 샌드박스(120B)) 내에서 실행된다. 네이티브 클라이언트 호환성 머신 코드는 브라우저와 같은 애플리케이션 내에서 실행가능하다. 게스트 소프트웨어를 샌드박스 내에서의 실행을 위해 네이티브 클라이언트 호환성 머신 코드로 이진 변환하는 것을 제공하는 것은, 에뮬레이팅된 게스트 소프트웨어의 런타임 동안, 그리고 게스트 소프트웨어의 포팅 또는 리컴파일링 없이 저스트 인 타임으로 발생할 수 있다. 이진 변환을 제공하는 것은, 에뮬레이팅된 게스트 소프트웨어의 실행과 인터리빙된다.

[0041] [0045] 게스트 소프트웨어는, 예를 들어, ARM 하드웨어 시스템 또는 x86 하드웨어 시스템에서 실행되도록 설계된, ARM 소프트웨어 또는 x86 소프트웨어를 포함할 수 있다. 게스트 소프트웨어의 일부 코드는, 네이티브 클라이언트 코드 또는 네이티브 클라이언트 환경과 호환가능한 코드로 대체될 수 있다. 게스트 소프트웨어는, 일부 경우들에서, 안전하지 않거나 또는 포팅가능하지 않을 수 있다. 네이티브 클라이언트 호환성 머신 코드는 안전하고 포팅가능할 수 있다. 게스트 소프트웨어는, 특정된 운영 시스템 내에서만 실행되도록 구성될 수 있다. 네이티브 클라이언트 환경은 다수의 운영 시스템들 중 임의의 것에서 실행이 가능할 수 있다.

[0042] [0046] 게스트 소프트웨어를 네이티브 클라이언트 호환성 머신 코드로 이진 변환하는 것을 제공하는 것은, 특정된 게스트 하드웨어 아키텍처에서 실행될 경우 게스트 소프트웨어에 의해 사용되는 레지스터들(예를 들어, 게스트 레지스터 들108A))을 나타내는 가상 레지스터들(예를 들어, 에뮬레이팅된 게스트 레지스터들(108B))의 세트를 생성하는 것을 포함할 수 있다. 가상 레지스터들의 어드레스들은 베이스 포인터(RBP) 플러스 미리결정된 오프셋에 의해 참조될 수 있다. RBP 플러스 오프셋 기술의 결과로서, 각각의 레지스터는, 하나의 명령을 통해, 샌드박스 내에서부터 액세스가능할 수 있다. 예를 들어, 레지스터 R0은 어드레스 RBP에 있을 수 있고, 레지스터 R1은 어드레스 RBP + 4에 있을 수 있고, 레지스터 R2는 어드레스 RBP + 8에 있을 수 있고, 레지스터 Rn은 어드레스 RBP + 4n에 있을 수 있는 식이다(여기서, n은 정수이다).

[0043] [0047] 게스트 소프트웨어의 네이티브 클라이언트 호환성 머신 코드로의 이진 변환을 제공하는 것은 게스트 하드웨어 아키텍처의 특징들 및 게스트 소프트웨어의 API 호들을 네이티브 클라이언트 내에서 에뮬레이팅하는 것을 포함할 수 있다. 게스트 하드웨어 아키텍처로부터의 시스템 API들 또는 벤더 API들은 네이티브 클라이언트 샌드박스 내에서 실행되도록 이진 변환될 수 있다. 대안으로, 게스트 하드웨어 아키텍처의 시스템 API들 중 일부가 네이티브 클라이언트 샌드박스의 시스템 API들로 대체될 수 있다.

[0044] [0048] 이진 변환 동안, 게스트 소프트웨어는 네이티브 클라이언트의 규칙들과 호환가능한 포맷으로 변환될 수 있다. 예를 들어, 네이티브 클라이언트는, 32로 제산될 수 있는 어드레스들과 교차하는 명령들이 제공되지 않을 것을 요구한다. 이러한 경우들에서, 32로 제산가능한 어드레스들은 "번들 경계"로 지칭될 수 있다. 에뮬레이터에 의해 생성된 모든 명령들은, 명령들이 번들 경계와 결코 교차하지 않으며, NOP(no-operation) 명령들로 채워지는 그러한 방식으로 방출된다. 네이티브 클라이언트 환경을 위한 샌드박스는, 호스트 머신에서 이용가능한 레지스터들에 또는 메모리에 저장된 에뮬레이팅된 게스트 레지스터들의 세트에 액세스한다. 네이티브 클라이언트 호환성 머신 코드와 연관되는 데이터는 네이티브 클라이언트 환경을 위한 샌드박스 내에 저장된다. 에뮬레이팅된 게스트 레지스터들은, 특정된 게스트 하드웨어 아키텍처의 레지스터들에 해당한다.

[0045] [0049] 단계(230)에서, 호스트 머신은 네이티브 클라이언트 환경에서 그리고 샌드박스 내에서 네이티브 클라이언트 호환성 머신 코드를 실행한다. 네이티브 클라이언트 호환성 머신 코드는 네이티브 클라이언트 샌드박스 내의 레지스터들 및 데이터에 액세스할 수 있지만, 네이티브 클라이언트 샌드박스 외부의 레지스터들 또는 데이터에는 액세스하지 않을 수 있다. 샌드박싱된 소프트웨어는 호스트의 레지스터들에 또는 메모리에 임시로 저장되는 에뮬레이팅된 게스트 레지스터들에 액세스한다. 단계(230) 이후, 프로세스(200)가 종료된다.

[0046] [0050] 상술된 바와 같이, 프로세스(200)의 단계들(210-230)은 특정 순서에 따라 그리고 연속하여 구현된다. 그러나, 단계들(210-230)은 임의의 순서로 구현될 수 있다. 일부 실시예들에서, 단계들(210-230) 중 2개 또는 그 초과의 것이 동시에 구현될 수 있다.

[0047] [0051] 도 3은 본 기술의 일부 구현들이 구현되는 전자 시스템(300)을 개념적으로 예시한다. 예를 들어, 게스트 시스템(100A) 또는 호스트 머신(100B) 중 하나 또는 그 초과의 것은 전자 시스템(300)의 어레인지먼트를 사용하여 구현될 수 있다. 전자 시스템(300)은 컴퓨터(예를 들어, 모바일 전화, PDA), 또는 임의의 다른 종류의 전자식 디바이스일 수 있다. 이러한 전자 시스템은 다양한 타입들의 컴퓨터 판독가능 매체들 및 다양한 다른 타입들의 컴퓨터 판독가능 매체들을 위한 인터페이스들을 포함한다. 전자 시스템(300)은 버스(305), 프로세서(들)(310), 시스템 메모리(315), 판독-전용 메모리(320), 영구 저장 디바이스(325), 입력 디바이스 인터페이스(330), 출력 디바이스 인터페이스(335), 및 네트워크 인터페이스(340)를 포함한다.

[0048] [0052] 버스(305)는 전자 시스템(300)의 수 많은 내부 디바이스들을 통신가능하게 연결하는 모든 시스템, 주변기기 및 칩셋 버스들을 총괄적으로 나타낸다. 예를 들어, 버스(305)는 프로세서(들)(310)을 판독-전용 메모리(320), 시스템 메모리(315) 및 영구 저장 디바이스(325)와 통신가능하게 연결한다.

[0049] [0053] 이들 다양한 메모리 유닛들로부터, 프로세서(들)(310)는, 본 기술의 프로세스들을 실행하기 위해 프로세싱할 데이터 및 실행할 명령들을 리트리빙한다. 프로세서(들)는 상이한 구현들에서 하나의 프로세서 또는 멀티-코어 프로세서를 포함할 수 있다.

[0050] [0054] ROM(read-only-memory)(320)은, 프로세서(들)(310) 및 전자 시스템의 다른 모듈들에 의해 필요로 되는 정적 데이터 및 명령들을 저장한다. 영구 저장 디바이스(325)는, 다른 한편으로, 판독 및 기록 메모리 디바이스이다. 이 디바이스는 전자 시스템(300)이 오프인 경우에도 명령들 및 데이터를 저장하는 비휘발성 메모리 유닛이다. 본 기술의 일부 구현들은, 영구 저장 디바이스(325)로서 대용량 저장 디바이스(예를 들어, 자기 또는 광학 디스크 및 그의 대응하는 디스크 드라이브)를 사용한다.

[0051] [0055] 다른 구현들은 영구 저장 디바이스(325)로서 착탈식 저장 디바이스(예를 들어, 플로피 디스크, 플래시 드라이브 또는 디스크 드라이브)를 사용한다. 영구 저장 디바이스(325)와 같이, 시스템 메모리(315)는 판독-및-기록 메모리 디바이스이다. 그러나, 저장소 디바이스(325)와 다르게, 시스템 메모리(315)는 휘발성 판독-및-기록 메모리, 그러한 랜덤 액세스 메모리이다. 시스템 메모리(315)는 프로세서가 런타임 시 필요로 하는 명령들 및 데이터 중 일부를 저장한다. 일부 구현들에서, 본 기술의 프로세스들이 시스템 메모리(315), 영구 저장 디바이스(325), 또는 판독-전용 메모리(320)에 저장된다. 예를 들어, 다양한 메모리 유닛들은 일부 구현들에 따라 네이티브 클라이언트로 이진 전환하기 위한 명령들을 포함한다. 이러한 다양한 메모리 유닛들로부터, 프로세서(들)(310)는 일부 구현들의 프로세스들을 실행하기 위해서 실행할 명령들 및 프로세싱할 데이터를 리트리빙한다.

[0052] [0056] 버스(305)는 또한 입력 및 출력 디바이스 인터페이스들(330 및 335)에 연결한다. 입력 디바이스 인터페이스(330)는 사용자가 정보를 통신하고 전자 시스템에 대한 커맨드들을 선택하게 할 수 있다. 입력 디바이스 인터페이스(330)와 사용되는 입력 디바이스들은, 예를 들어, 문자숫자식 키보드들 및 포인팅 디바이스들("커서 제어 디바이스들"로도 지칭됨)을 포함한다. 출력 디바이스 인터페이스들(335)은, 예를 들어, 전자 시스템(300)에 의해 생성되는 이미지들의 디스플레이를 가능하게 한다. 출력 디바이스 인터페이스(335)와 사용되는 출력 디바이스들은, 예를 들어, 프린터들 및 디스플레이 디바이스들, 예를 들어 CRT(cathode ray tubes) 또는 LCD(liquid crystal displays)를 포함한다. 일부 구현들은 디바이스들, 예를 들어 입력 및 출력 디바이스들 양쪽 모두로서 기능하는 터치 스크린을 포함한다.

[0053] [0057] 마지막으로, 도 3에 도시된 바와 같이, 버스(305)는 또한 네트워크 인터페이스(340)를 통해 전자 시스템(300)을 네트워크(미도시)에 결합한다. 이러한 방식으로, 전자 시스템(300)은 컴퓨터들의 네트워크(예를 들어, LAN(local area network), WAN(wide area network), 또는 인트라넷), 또는 네트워크들의 네트워크, 예를 들어 인터넷의 일부일 수 있다. 전자 시스템(300)의 임의의 또는 모든 컴포넌트들은 본 기술과 연계하여 사용될 수

있다.

[0054] [0058] 위에서 설명된 특징들 및 애플리케이션들은 컴퓨터 판독가능 저장 매체(컴퓨터 판독가능 매체로도 지칭됨) 상에 레코딩되는 명령들의 세트로서 특정되는 소프트웨어 프로세스들로서 구현될 수 있다. 이러한 명령들이 하나 또는 그 초과의 프로세서(들)(예를 들어, 하나 또는 그 초과의 프로세서들, 프로세서들의 코어들, 또는 다른 프로세싱 유닛들)에 의해 실행될 때, 그들은 프로세서(들)로 하여금 명령들에 표시된 동작들을 수행하게 한다. 컴퓨터 판독가능 매체들의 예들은 CD-ROM들, 플래시 드라이브들, RAM 칩들, 하드 드라이브들, EPROM들 등을 포함하지만 이들로 제한되지는 않는다. 컴퓨터 판독가능 매체들은 무선으로 또는 유선 연결들을 통해 전달되는 캐리어 파들 및 전자 신호들을 포함하지 않는다.

[0055] [0059] 본 상세한 설명에서, 용어 "소프트웨어"는 판독-전용 메모리에 상주하는 펌웨어, 또는 프로세서에 의한 프로세싱을 위해 메모리로 판독될 수 있는 자기 저장부 또는 플래시 저장부, 예를 들어 고체-상태 드라이브에 저장된 애플리케이션들을 포함하도록 의도된다. 또한, 일부 구현들에서, 별개의 소프트웨어 기술들을 유지하면서 다수의 소프트웨어 기술들이 더 큰 프로그램의 서브-부분들로서 구현될 수 있다. 일부 구현들에서, 다수의 소프트웨어 기술들은 또한 별도의 프로그램들로 구현될 수 있다. 마지막으로, 여기서 설명된 소프트웨어 기술을 함께 구현하는 별도의 프로그램들의 임의의 조합은 본 기술의 범위 내에 있다. 일부 구현들에서, 소프트웨어 프로그램은, 하나 또는 그 초과의 전자 시스템들에서 동작하도록 설치될 때, 소프트웨어 프로그램들의 동작들을 실행하고 수행하는 하나 또는 그 초과의 특정 머신 구현들을 정의한다.

[0056] [0060] 컴퓨터 프로그램(프로그램, 소프트웨어, 소프트웨어 애플리케이션, 스크립트 또는 코드로도 공지됨)은 컴파일링되거나 해석되는 언어들, 선언형(declarative) 또는 절차형(procedural) 언어들을 비롯한 임의의 형태의 프로그래밍 언어로 기록될 수 있고, 그것은 독립형 프로그램이나 또는 모듈, 컴포넌트, 서브루틴, 오브젝트 또는 컴퓨팅 환경에서 사용하기 위해 적합한 다른 유닛을 비롯한 임의의 형태로 전개될 수 있다. 컴퓨터 프로그램은 파일 시스템의 파일에 대응할 수 있지만 반드시 그런 것은 아니다. 프로그램은 다른 프로그램들 또는 데이터(예를 들어, 마크업 언어 문헌에 저장되는 하나 또는 그 초과의 스크립트들)을 보유하는 파일의 부분에, 해당 프로그램에 전용화된 하나의 파일에, 또는 다수의 통합형 파일들(예를 들어, 하나 또는 그 초과의 모듈들, 서브 프로그램들 또는 코드의 부분들을 저장하는 파일들)에 저장될 수 있다. 컴퓨터 프로그램은 하나의 사이트에 위치되거나 다수의 사이트들에 걸쳐 분산되어 통신 네트워크에 의해 상호연결되는 다수의 컴퓨터들에서 실행되거나 또는 하나의 컴퓨터에서 실행되도록 전개될 수 있다.

[0057] [0061] 위에서 설명된 기능들은 디지털 전자 회로에서, 컴퓨터 소프트웨어, 펌웨어 또는 하드웨어에서 구현될 수 있다. 기술들은 하나 또는 그 초과의 컴퓨터 프로그램 제품들을 사용하여 구현될 수 있다. 프로그램가능 프로세서들 및 컴퓨터들은 모바일 디바이스들에 포함되거나 또는 모바일 디바이스들로서 패키징될 수 있다. 프로세스들 및 로직 흐름들은 하나 또는 그 초과의 프로그램가능 프로세서들에 의해서 및 하나 또는 그 초과의 프로그램가능 로직 회로에 의해서 수행될 수 있다. 범용의 및 특수 용도의 컴퓨팅 디바이스들 및 저장 디바이스들이 통신 네트워크들을 통해 상호연결될 수 있다.

[0058] [0062] 일부 구현들은 전자 컴포넌트들, 예를 들어 머신-판독가능 또는 컴퓨터-판독가능 매체(대안적으로는 컴퓨터-판독가능 저장 매체들, 머신-판독가능 매체들 또는 머신-판독가능 저장 매체들로 지칭됨)에 컴퓨터 프로그램 명령들을 저장하는 메모리, 저장부 및 마이크로프로세서들을 포함한다. 그러한 컴퓨터-판독가능 매체들의 일부 예들은 RAM, ROM, CD-ROM(read-only compact discs), CD-R(recordable compact discs), CD-RW(rewritable compact discs), 판독-전용 디지털 다기능 디스크들(예를 들어, DVD-ROM, 듀얼-층 DVD-ROM), 다양한 레코딩가능/재기록가능 DVD들(예컨대, DVD-RAM, DVD-RW, DVD+RW 등), 플래시 메모리(예를 들어, SD 카드들, 미니-SD 카드들, 마이크로-SD 카드들 등), 자기 또는 고체 상태 하드 드라이브들, 판독-전용 및 레코딩가능 Blu-Ray® 디스크들, 울트라-밀도 광학 디스크들, 임의의 다른 광학 또는 자기 매체들, 및 플로피 디스크들을 포함한다. 컴퓨터-판독가능 매체들은, 적어도 하나의 프로세서에 의해 실행가능하고 다양한 동작들을 수행하기 위한 명령들의 세트들을 포함하는 컴퓨터 프로그램을 저장할 수 있다. 컴퓨터 프로그램들 또는 컴퓨터 코드의 예들은 예컨대 컴파일러에 의해 생성되는 머신 코드, 및 해석기를 사용하여 컴퓨터, 전자 컴포넌트 또는 마이크로프로세서에 의해 실행되는 더 고 레벨의 코드를 포함하는 파일들을 포함한다.

[0059] [0063] 상기 논의는 소프트웨어를 실행하는 마이크로프로세서 또는 멀티-코어 프로세서들을 주로 지칭하지만, 일부 구현들은 하나 또는 그 초과의 집적 회로들, 예를 들어 ASIC들(application specific integrated circuits) 또는 FPGA들(field programmable gate arrays)에 의해 수행된다. 일부 구현들에서, 그러한 집적 회로들은 그 회로 자체에 저장되는 명령들을 실행한다.

[0060] [0064] 본 출원의 이러한 상세한 설명 및 임의의 청구항에서 사용되는 바와 같이, 용어들 "컴퓨터", "서버", "프로세서" 및 "메모리" 모두는 전자 또는 다른 기술적 디바이스들을 지칭한다. 이러한 용어들은 사람들 또는 사람들의 그룹들은 배제한다. 상세한 설명을 위해서, 용어들 디스플레이 또는 디스플레하는 것은 전자 디바이스에 디스플레이하는 것을 의미한다. 본 출원의 이러한 상세한 설명 및 임의의 청구항에서 사용되는 바와 같이, 용어들 "컴퓨터 판독가능 매체" 및 "컴퓨터 판독가능 매체들"은 전적으로, 컴퓨터에 의해 판독가능한 형태로 정보를 저장하는 유형의(tangible) 물리적 오브젝트들로 제약된다. 이러한 용어들은 임의의 무선 신호들, 유선 다운로드 신호들 및 임의의 다른 임시(ephemeral) 신호들은 배제한다.

[0061] [0065] 사용자와의 상호작용을 제공하기 위해, 이러한 상세한 설명에 설명된 청구대상의 구현들은, 사용자에게 정보를 디스플레이하기 위한 디스플레이 디바이스, 예를 들어 CRT(cathode ray tube) 또는 LCD(liquid crystal display) 모니터, 및 사용자가 컴퓨터에 입력을 제공할 수 있게 하는 키보드 및 포인팅 디바이스, 예를 들어 마우스 또는 트랙볼을 갖는 컴퓨터에서 구현될 수 있다. 다른 종류들의 디바이스들이 또한 사용자와의 상호작용을 제공하기 위해 사용될 수 있다; 예를 들어, 사용자에게 제공되는 피드백은 감각 피드백, 예를 들면, 시각 피드백, 청각 피드백, 또는 촉각 피드백의 임의의 형태일 수 있으며; 사용자로부터의 입력은 음향, 스피치, 또는 촉각 입력을 포함하는 임의의 형태로 수신될 수 있다. 또한, 컴퓨터는, 사용자에 의해 사용되는 디바이스에 문헌들을 전송하고 이 디바이스로부터 문헌들을 수신함으로써; 예를 들어 사용자의 클라이언트 디바이스 상의 웹 브라우저로부터 수신되는 요청들에 대한 응답으로 그 웹 브라우저에 웹 페이지들을 전송함으로써 사용자와 상호작용할 수 있다.

[0062] [0066] 이러한 상세한 설명에 설명된 청구대상은 컴퓨팅 시스템에서 구현될 수 있는데, 그 컴퓨팅 시스템은, 예를 들어 데이터 서버로서 백 엔드(back end) 컴포넌트를 포함하거나, 미들웨어(mfiddleware) 컴포넌트, 예를 들어 애플리케이션 서버를 포함하거나, 또는 프론트 엔드(front end) 컴포넌트, 예를 들어, 이러한 상세한 설명에 설명된 청구대상의 구현과 사용자가 상호작용할 수 있게 하는 웹 브라우저 또는 그래픽 사용자 인터페이스를 갖는 클라이언트 컴퓨터를 포함하거나, 또는 이러한 백 엔드, 미들웨어 또는 프론트 엔드 컴포넌트들 중 하나 또는 그 초과의 것의 임의의 조합을 포함한다. 시스템의 컴포넌트들은 임의의 형태 또는 매체의 디지털 데이터 통신, 예를 들어, 통신 네트워크에 의해 상호연결될 수 있다. 통신 네트워크들의 예들은, LAN(local area network) 및 WAN(wide area network), 인터-네트워크(예를 들어, 인터넷) 및 피어-투-피어 네트워크들(예를 들어, 애드혹 피어-투-피어 네트워크들)을 포함한다.

[0063] [0067] 컴퓨팅 시스템은 클라이언트들 및 서버들을 포함할 수 있다. 클라이언트 및 서버는 일반적으로 서로 멀리 떨어져 있으며, 통상적으로 통신 네트워크를 통해 상호작용한다. 클라이언트와 서버의 관계는 개별 컴퓨터들에서 실행되며 서로 간에 클라이언트-서버 관계를 갖는 컴퓨터 프로그램들을 통해 발생한다. 개시된 청구대상의 일부 양상들에서, 서버는 (예를 들어, 클라이언트 디바이스와 상호작용하는 사용자에게 데이터를 디스플레이하고 사용자로부터 사용자 입력을 수신하기 위해) 클라이언트 디바이스에 데이터(예를 들어, HTML 페이지)를 송신한다. 클라이언트 디바이스에서 생성되는 데이터(예를 들어, 사용자 상호작용의 결과)가 서버의 클라이언트 디바이스로부터 수신될 수 있다.

[0064] [0068] 개시된 프로세스들에서 단계들의 임의의 특정 순서 또는 계층은 예시적인 접근법의 예시라는 것을 이해한다. 설계 선호도들에 기초하여, 프로세스들에서의 단계들의 특정 순서 또는 계층이 재배열될 수 있다는 것 또는 모든 예시된 단계들이 수행될 수 있다는 것을 이해한다. 단계들 중 일부는 동시적으로 수행될 수 있다. 예를 들어, 특정 환경들에서는, 다중작업 및 병렬 프로세싱이 유리할 수 있다. 더욱이, 위에서 예시된 다양한 시스템 컴포넌트들의 분리는 그러한 분리를 필요로 하는 것으로 이해되지 않아야 하고, 설명된 프로그램 컴포넌트들 및 시스템들은 일반적으로 하나의 소프트웨어 제품에 함께 통합되거나 또는 다수의 소프트웨어 제품들로 패키징될 수 있다는 것을 이해해야 한다.

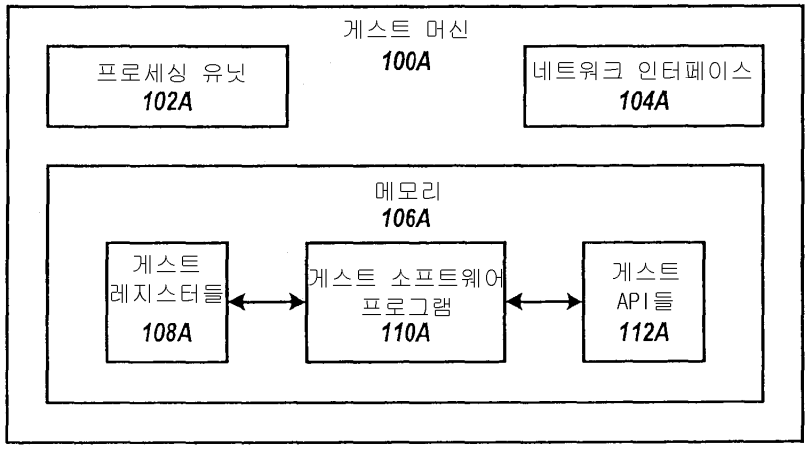
[0065] [0069] 이러한 양상들의 다양한 수정들이 쉽게 자명할 것이고, 본원에 정의된 일반적인 원리들이 다른 양상들에 적용될 수 있다. 따라서, 청구항들은 본원에 제시된 양상들로 제한되도록 의도되지 않고, 청구항 문언에 부합하는 전체 범위로 제공될 것이고, 단수 형태의 엘리먼트에 대한 참조는 특별히 그렇게 설명되지 않는 한은 "하나 및 단지 하나"를 의미하도록 의도되지 않고 오히려 "하나 또는 그 초과"를 의미하도록 의도된다. 구체적으로 달리 설명되지 않는 한, "일부"라는 용어는 하나 또는 그 초과를 지칭한다. 남성의 대명사(예를 들어, 그의)는 여성 및 중성 성별(예를 들어, 그녀의 및 그것의)을 포함하고, 그 반대의 경우도 가능하다. 제목들(headings) 및 부제들(subheadings)은, 만약 있다면, 단지 편의상 사용되며 본 기술을 제한하지는 않는다.

[0066] [0070] 문구, 예를 들어 "양상"은, 그 양상이 본 기술에 본질적이라는 것 또는 그 양상이 본 기술의 모든 구성

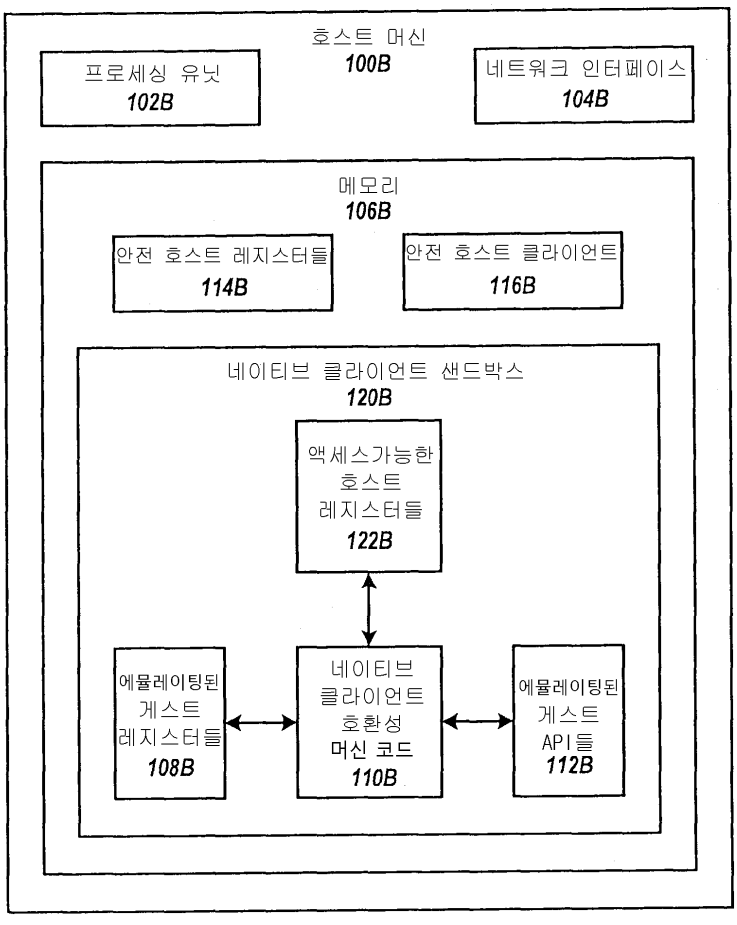
들에 적용된다는 것을 의미하지는 않는다. 양상에 관한 개시내용은 모든 구성들, 또는 하나 또는 그 초과의 구성들에 적용될 수 있다. 문구, 예를 들어 양상은 하나 또는 그 초과의 양상들을 지칭할 수 있고, 그 반대의 경우도 가능하다. 문구, 예를 들어 "구성"은, 그러한 구성이 본 기술에 본질적이라는 것 또는 그러한 구성이 본 기술의 모든 구성들에 적용된다는 것을 의미하지는 않는다. 구성에 관한 개시내용은 모든 구성들, 또는 하나 또는 그 초과의 구성들에 적용될 수 있다. 문구, 예를 들어 "구성"은 하나 또는 그 초과의 구성들을 지칭할 수 있고, 그 반대의 경우도 가능하다.

도면

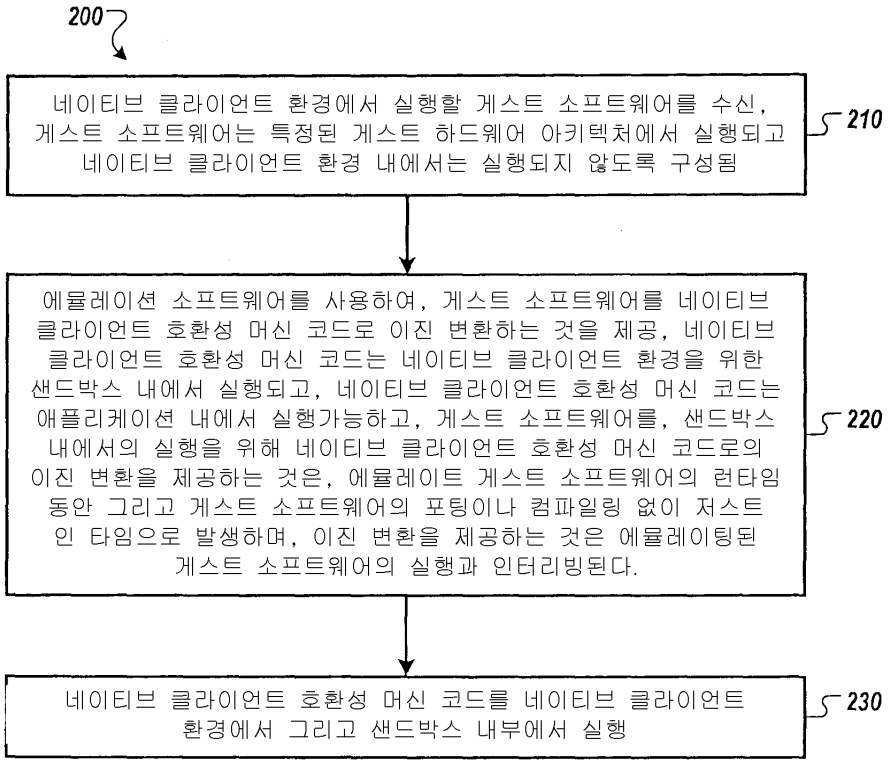
도면1a



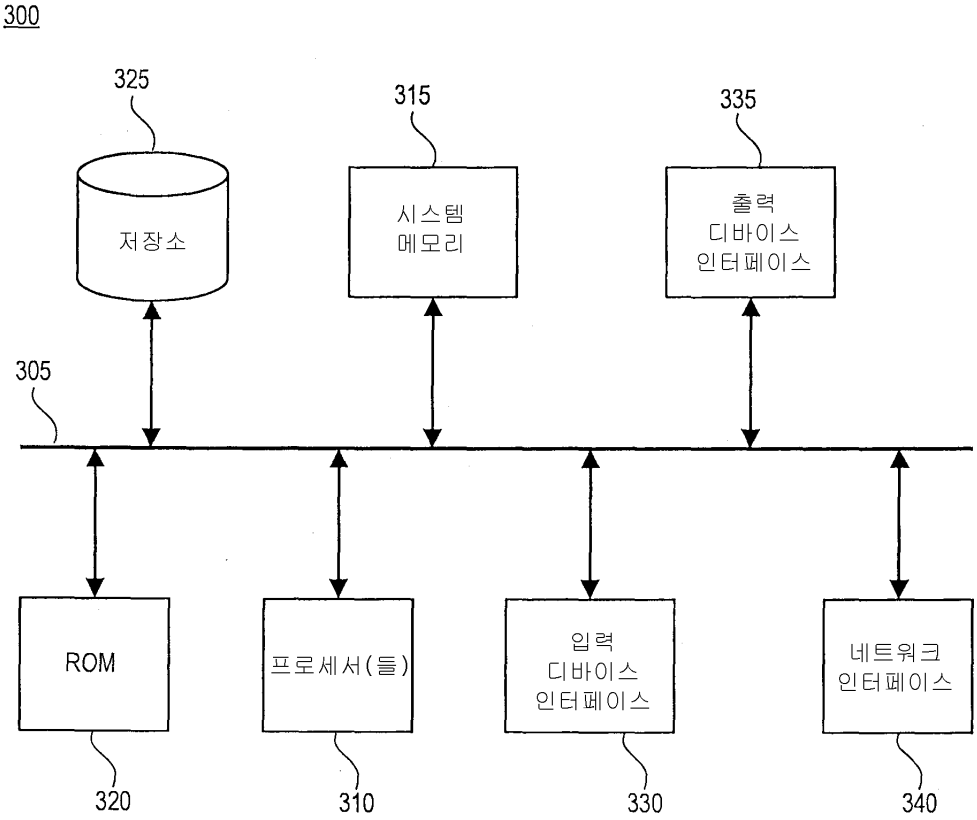
도면1b



도면2


200
네이티브 클라이언트 환경에서 실행할 게스트 소프트웨어를 수신, 게스트 소프트웨어는 특정된 게스트 하드웨어 아키텍처에서 실행되고 네이티브 클라이언트 환경 내에서는 실행되지 않도록 구성됨
210
에뮬레이션 소프트웨어를 사용하여, 게스트 소프트웨어를 네이티브 클라이언트 호환성 머신 코드로 이진 변환하는 것을 제공, 네이티브 클라이언트 호환성 머신 코드는 네이티브 클라이언트 환경을 위한 샌드박스 내에서 실행되고, 네이티브 클라이언트 호환성 머신 코드는 애플리케이션 내에서 실행가능하고, 게스트 소프트웨어를, 샌드박스 내에서의 실행을 위해 네이티브 클라이언트 호환성 머신 코드로의 이진 변환을 제공하는 것은, 에뮬레이트 게스트 소프트웨어의 런타임 동안 그리고 게스트 소프트웨어의 포팅이나 컴파일링 없이 저스트 인 타임으로 발생하며, 이진 변환을 제공하는 것은 에뮬레이팅된 게스트 소프트웨어의 실행과 인터리빙된다.
220
네이티브 클라이언트 호환성 머신 코드를 네이티브 클라이언트 환경에서 그리고 샌드박스 내부에서 실행
230

도면3


300
325
저장소
315
시스템 메모리
335
출력 디바이스 인터페이스
305
ROM
320
프로세서(들)
310
입력 디바이스 인터페이스
330
네트워크 인터페이스
340