(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2017-0129866
(43) 공개일자 2017년11월27일

(51) 국제특허분류(Int. Cl.)
H04L 9/32 (2006.01) *G06Q 20/06* (2012.01)
G06Q 20/38 (2012.01) *H04L 29/06* (2006.01)
H04W 12/06 (2009.01)
(52) CPC특허분류
H04L 9/3234 (2013.01)
G06Q 20/0655 (2013.01)
(21) 출원번호 10-2017-7030054
(22) 출원일자(국제) 2016년03월18일
심사청구일자 없음
(85) 번역문제출일자 2017년10월18일
(86) 국제출원번호 PCT/US2016/023142
(87) 국제공개번호 WO 2016/154001
국제공개일자 2016년09월29일
(30) 우선권주장
62/136,340 2015년03월20일 미국(US)
62/136,385 2015년03월20일 미국(US)

(71) 출원인
리베츠 코프.
미국 델라웨어 윌밍턴 오렌지 스트리트 1209 (우: 19801)
(72) 발명자
스프라그, 마이클
미국 10014 뉴욕 뉴욕 베드포드 스트리트 73
스프라그, 스티븐
미국 02154 매사추세츠 리치몬드 스웜프 로드 111
(74) 대리인
특허법인 남앤드남

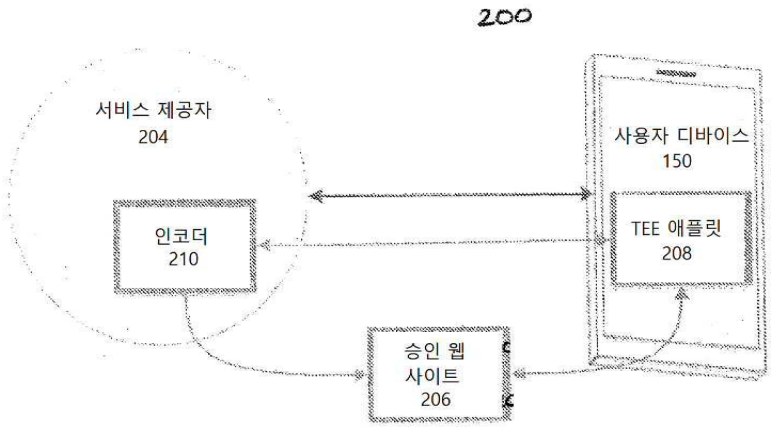
전체 청구항 수 : 총 17 항

(54) 발명의 명칭 **블록 체인을 사용하여 디바이스 무결성의 자동화된 입증**

(57) 요 약

블록 체인 트랜잭션들에 대한 보안을 추가로 제공할 블록 체인 트랜잭션의 수락 이전에 알려지지 않은 클라이언트 디바이스의 전체 입증을 제공하는 시스템들 및 방법들이 개시된다. 디바이스의 헬스는 전자 트랜잭션들에 관여하기 이전에 입증될 수 있다. 일부 실시예들에서, 전체 디바이스 무결성 검증의 자동화가 블록 체인 트랜잭션의 일부로서 제공된다. 본 발명의 특정 양태들은 디바이스들의 신뢰를 가능하게 한다. 일부 실시예는 디바이스와의 신뢰 가능한 관계가 최종 사용자와의 훨씬 더 안전하고, 더 용이하고, 더 강한 관계에 기여할 수 있다는 기본적 전제 상에서 운용된다. 이를 성취하는 것은 현재의 트랜잭션에 수반되는 디바이스가 앞선 트랜잭션들에서의 것과 동일한 디바이스라는 확신을 인지하는 것을 필요로 한다.

대 표 도 - 도2a



본 발명에 따른 예시적 디바이스 승인 시스템

(52) CPC특허분류
G06Q 20/3829 (2013.01)
H04L 63/0823 (2013.01)
H04L 63/126 (2013.01)
H04W 12/06 (2013.01)
G06Q 2220/00 (2013.01)
H04L 2209/127 (2013.01)
H04L 2209/56 (2013.01)

명 세 서

청구범위

청구항 1

블록 체인 통신 네트워크에서의 사용자 디바이스의 디바이스 무결성을 검증하는 컴퓨터 구현 방법으로서:

상기 블록 체인 네트워크에서의 전자 트랜잭션을 전달하는 것의 대비로, 상기 트랜잭션의 일부로서 디바이스 무결성 검증 프로세스를 구현하는 단계로서:

상기 사용자 디바이스에서의 신뢰의 루트로부터 디바이스 실행 환경의 무결성의 내부 입증을 수행하는 단계; 및

전자 시그니처를 필요로 하여, 상기 시그니처의 무결성의 검증이 블록 체인 트랜잭션에 적용되는 단계로서;

상기 시그니처의 무결성의 검증은 상기 디바이스의 상기 실행 환경이 알려진 양호한 조건으로 있는지 여부의 판단에 기반하며:

상기 시그니처의 무결성에 기반하여, 상기 트랜잭션이 진행되는 것을 가능하게 하거나, 상기 디바이스의 상기 실행 환경이 알려진 양호한 조건으로 있지 않다는 것이 판단되더라도 사용자에 의해 의도되는 바에 따라 상기 전자 트랜잭션이 진행되는 것이 가능해지는 것을 검증하기 위해 개선 권한을 요청하는 단계를 포함하는 단계를 포함하는 단계를 포함하는, 방법.

청구항 2

제1항에 있어서,

상기 시그니처의 무결성의 검증은:

처리를 위해 상기 블록 체인 네트워크에 신뢰 명령어의 루트를 송신하여, 상기 블록 체인 네트워크의 적어도 일부가 상기 전자 트랜잭션을 수락하기 위해 다수의 전자 시그니처를 필요로 함으로써 응답하는 단계로서:

상기 디바이스의 상기 실행 환경 내에서, 상기 사용자 디바이스에서의 신뢰의 루트로부터 명령어를 생성하는 단계;

상기 신뢰 명령어의 루트에 상응하는 제1 전자 시그니처를 필요로 하여, 상기 시그니처의 무결성의 검증이 상기 블록 체인 트랜잭션에 적용되는 단계; 및

상기 디바이스의 상기 실행 환경이 알려진 양호한 조건으로 있는지 여부의 판단에 기반하여 상기 시그니처의 무결성을 검증함으로써 상기 제1 전자 시그니처에 응답하는 단계로서:

상기 시그니처를 앞서 기록된 참조값과 비교하는 단계;

상기 시그니처가 상기 앞서 기록된 참조값에 부합하면, 그 때 상기 트랜잭션이 진행되는 것을 가능하게 하는 단계; 및

상기 시그니처가 상기 앞서 기록된 참조값에 부합하지 않으면, 상기 디바이스의 상기 실행 환경이 알려진 양호한 조건으로 있지 않다는 것이 판단되더라도 상기 사용자에 의해 의도되는 바에 따라 상기 전자 트랜잭션이 진행되는 것이 가능해지는 것을 검증하기 위해 제3 자 대역 외 프로세스를 요청하는 단계를 포함하는 단계를 포함하는 단계를 포함하는, 방법.

청구항 3

제1항에 있어서,

상기 시그니처의 무결성을 검증하는 단계는:

상기 디바이스의 상기 실행 환경이 알려진 양호한 조건으로 있는지 여부의 판단에 기반하여 상기 디바이스가 상기 전자 시그니처를 제공하는 단계:

상기 디바이스가 상기 전자 시그니처를 제공하면, 상기 트랜잭션이 진행되는 것을 가능하게 하는 단계;

상기 개선 권한이 상기 시그니처를 제공하면, 상기 디바이스의 상기 실행 환경이 알려진 양호한 조건으로 있지 않다는 것이 판단되더라도 상기 사용자에 의해 의도되는 바에 따라 상기 트랜잭션이 진행되는 것을 가능하게 하는 단계를 포함하는, 방법.

청구항 4

제2항에 있어서,

상기 대역 외 프로세스는: 상기 사용자의 의도가 미리 결정된 필요 조건들을 충족시키거나, 디바이스 무결성이 미리 결정된 필요 조건들을 충족시키거나, 부가 프로세스가 미리 결정된 필요 조건들을 충족시킨다는 것 중 적어도 하나를 확인하기 위해 N 또는 M 암호화 키 기능을 사용하는 단계를 더 포함하는, 방법.

청구항 5

제2항에 있어서,

상기 참조값은 디바이스 플랫폼의 소유주에 의해 수행되는 등록 프로세스 동안 생성되는, 방법.

청구항 6

제2항에 있어서,

상기 참조값은 상기 디바이스에 할당되는 버스 증명서에 기반하여 생성되며, 상기 버스 증명서는 상기 디바이스의 제조자 또는 생성자, 상기 디바이스의 상기 실행 환경의 제조자 또는 생성자, 및/또는 상기 디바이스 상의 애플리케이션의 제조자 또는 생성자에 의해 생성되는, 방법.

청구항 7

제2항에 있어서,

상기 참조값은 상기 디바이스의 제조자 또는 생성자, 상기 디바이스의 상기 실행 환경의 제조자 또는 생성자, 및/또는 상기 디바이스 상의 애플리케이션의 제조자 또는 생성자 중 적어도 하나의 시그니처를 포함하는, 방법.

청구항 8

제2항에 있어서,

상기 제3 자 대역 외 프로세스는 상기 트랜잭션을 검증하기 위해 상기 요청에 응하여 토큰을 복귀시키는, 방법.

청구항 9

제2항에 있어서,

상기 시그니처가 상기 앞서 기록된 참조값에 부합하지 않으면, 상기 전자 트랜잭션이 일정 기간 내에 완료되는 것을 추가로 가능하게 하는, 방법.

청구항 10

제2항에 있어서,

상기 참조값의 등록과 상기 트랜잭션 사이의 기간 및/또는 상기 트랜잭션의 양에 기반하여 상기 디바이스의 상기 실행 환경이 알려진 양호한 조건으로 있지 않다는 것이 판단되더라도 상기 의도된 전자 트랜잭션이 진행되는 것이 가능해지는 것을 검증하는, 방법.

청구항 11

제10항에 있어서,

상기 기간이 미리 결정된 필요 조건들을 충족시키면, 임계량을 넘는 트랜잭션들이 진행되는 것이 가능해지는, 방법.

청구항 12

제11항에 있어서,

일정량을 넘는 트랜잭션을 가능하게 하는 것이 최소 수의 앞서 가능해진 트랜잭션에 기반하는, 방법.

청구항 13

제1항에 있어서,

디바이스 무결성이 최소 미리 결정된 필요 조건을 충족시키는지 여부 및 취해질 추가 작동들을 상기 사용자에게 나타내는 디스플레이 디바이스를 사용하는 단계를 더 포함하는, 방법.

청구항 14

제1항에 있어서,

제3 자에게의 상기 트랜잭션의 통지를 더 포함하며, 상기 통지에 응하여, 상기 제3 자는 상기 트랜잭션 및 상기 디바이스의 상태를 기록하는, 방법.

청구항 15

제14항에 있어서,

상기 제3 자는 상기 트랜잭션의 장래 분석을 위해 디바이스 무결성과 연관된 측정치들을 기록하는, 방법.

청구항 16

제14항에 있어서,

상기 기록이 인증된 제3 당사자들에게만 이용 가능해지도록 상기 기록을 암호로 혼란하게 하는 것을 포함하는 상기 기록의 프라이버시를 추가로 보장하는, 방법.

청구항 17

블록 체인 통신 네트워크에서의 사용자 디바이스의 디바이스 무결성을 검증하는 컴퓨터 구현 시스템으로서:

블록 체인 통신 네트워크;

상기 블록 체인 네트워크에서의 사용자 디바이스;

상기 블록 체인 네트워크에서의 전자 트랜잭션;

블록 체인 네트워크에서의 상기 전자 트랜잭션의 전달의 대비로 상기 트랜잭션의 일부로서 구현되는 디바이스 검증 프로세스를 포함하며, 상기 구현은:

상기 디바이스에서의 신뢰의 루트로부터 수행되는 디바이스 실행 환경의 무결성의 내부 입증;

상기 시그니처의 무결성의 검증이 블록 체인 트랜잭션에 적용되기 위한 전자 시그니처로서;

상기 시그니처의 무결성의 검증은 상기 디바이스의 상기 실행 환경이 알려진 양호한 조건으로 있는지 여부의 판단에 기반하며:

상기 시그니처의 무결성에 기반하여, 상기 트랜잭션이 진행되는 것을 가능하게 하거나, 상기 디바이스의 상기 실행 환경이 알려진 양호한 조건으로 있지 않다는 것이 판단되더라도 사용자에 의해 의도되는 바에 따라 상기 전자 트랜잭션이 진행되는 것이 가능해지는 것을 검증하기 위해 개선 권한을 요청하는 것을 포함하는 전자 시그니처를 더 포함하는, 컴퓨터 구현 시스템.

발명의 설명

기 술 분 야

[0001] 본 출원은 2015년 3월 20일자로 출원된 미국 가출원 제 62/136,340호 및 2015년 3월 20일자로 출원된 미국 가출원 제 62/136,385호의 이익을 주장한다. 위의 출원들의 전체 교시들은 참조로 본원에 포함된다.

배 경 기 술

[0002] 비트코인과 같은 분산화된 트랜잭션 시스템의 출현은 블록 체인으로서 알려져 있는 디지털값을 통해 소유권을 기록하는 신뢰할 수 있게 보안적인 프로토콜을 인터넷에 제공하였다. 시스템은 사람이 그러한 디지털값을 행사하는 것을 가능하게 하는 개인 키에 근원을 둔다. 그러나, 이러한 키가 디지털 방식으로 저장될 때, 그리고 특히 이러한 키가 트랜잭션될 때, 이러한 키는 실질적 손실을 야기할 수 있는 절도에 취약하다. 산업은 엔드포인트 디바이스에서의 높은 보장 작동에 대한 필요를 수년간 예상하였다. 이미 활용된 하드웨어 보안은 사람과 블록 체인 사이의 상호 작용에 대한 보안 및 프라이버시를 강화시키는데 사용될 수 있다.

[0003] 수천 개의 피어링된 서버의 후단에 의거하하는 공통 원장인, 비트코인 배후의 블록 체인은 수학적으로 불가해하도록 고안된다. 다수의 참여하는 피어가 커뮤니티의 지원으로 행하는 한은, 지난 값 그리고 따라서 절도값의 기록을 편집하기에 충분한 컴퓨팅 능력을 레버리징할 수 없다. 그러한 큰 커뮤니티의 무결성을 유지 관리하는 그러한 큰 커뮤니티로, 타원 곡선 암호화의 취약성만이 블록 체인을 위해할 수 있다는 것이 간주된다. 그러나, 블록 체인 그 자체가 양호하게 보안되지만, 개인이 블록 체인과 트랜잭션하는 방법은 매우 복잡하거나 다수의 널리 알려진 멀웨어 공격을 받기 쉽다. 결과는 블록 체인에 대한 명령어들의 품질이 보호된 트랜잭션 원장의 품질을 보장하는 것이 매우 중요하다는 것이다.

발명의 내용

해결하려는 과제

[0004] 비트코인 블록 체인에서 캡처되는 트랜잭션들의 대부분은 한 사람에서 다른 사람으로의 값의 전송을 기록한다. 공개 키들은 수반되는 당사자들을 나타낸다. 상응하는 개인 키들은 참여자가 상기 결과를 요청하는 것을 가능하게 한다. 감시 또는 제어의 어떤 다른 방법도 없으므로, 개인 키가 보안되는 것이 가장 중요하다. 블록 체인은 단명하는 구성체이다. 사람들은 네트워크 연결 디바이스의 사람들의 제어를 통해 블록 체인과 상호 작용할 수만 있다. 대체로 말하면, 이것이 일어나는 3가지 방식이 있다. A) 사람은 기계 자체가 피어이고 블록 체인으로 직접 기록하는 기계를 제어한다. B) 사람은 사람을 대신하여 행하는 서버에 명령하기 위해 웹 사이트 또는 모바일 앱을 사용하거나, C) 사람은 국부적으로 형성되는 트랜잭션을 전파하기 위해 웹 사이트 또는 앱을 사용한다.

[0005] 일반적으로, 개인 키는 요청에 서명하기 위해 적용된다. 실행 환경은 개인 키의 요청 및 보호의 정확성에 책임이 있다. 실행 환경의 헬스 및 근원에 대한 입증은 실행 환경의 신뢰성을 확립한다.

[0006] 실행 환경의 보안을 개선하기 위해 레버리징될 수 있는 다수의 광범위한 툴이 있다. 이는 범위가 하드웨어 후원 디바이스 신원에서 전체 신뢰되는 실행 환경들까지 이른다. 소비자 웹은 디바이스 신원 확인보다는 오히려 사용자 신원 확인 방법들 상에 구성되는 가장 폭넓게 분포된 서비스 플랫폼이다. 예를 들어, 서비스가 인에이블링(enabling) 디바이스에 의해 승인되는 모바일 전화 통신 또는 케이블 텔레비전과는 달리, 웹은 최종 사용자들이 신원 확인 프로토콜을 행하는 것, 즉 사용자 이름 및 비밀 번호를 입력하는 것을 필요로 한다. 이러한 방법의 휴대성에 대한 이익들이 있지만, 이는 실제로 위험하게 영향 받기 쉽다. 사용자들은 복잡한 비밀 번호들을 기억하는데 서투르고 반복되는 요청들에 의해 거슬려진다. 결과는 "Go Yanks" 같은 비밀 번호들 그리고 수일 동안 지속되는 것이 가능해지는 세션 키들이다. 다른 한편으로는, 디바이스는 디바이스의 하드웨어에 저장되는 수천 개의 크리덴셜 중 임의의 것을 갖는 임의의 사람의 용량을 훨씬 넘어서 암호화 승인에 만족스럽게 관여할 것이다. 그리고 디바이스는 지치는 것 없이 되풀이하여 그것을 행할 것이다.

[0007] 극도의 상황들에서를 제외하고, 사용자 이름/비밀 번호의 형태로의 휴대성은 임의의 역할을 갖는다. 그러나 대부분의 경우에서, 사용자들은 동일한 상호 작용들에 대해 동일한 디바이스들과 관계한다. 사용자들이 기본 승인을 행하기 위해 소유하는 디바이스들을 레버리징함으로써, 이러한 일관성은 사용자들에 대한 즉각적 액세스 그리고 서비스 제공자들에 대한 증가된 보장으로 보상될 수 있다.

[0008] 인터넷은 대체로 다목적 디바이스들에 의해 액세스된다. PC들, 태블릿들 및 전화기들은 수백 개의 애플리케이션을 호스팅할 수 있고 새로운 앱들에 대한 약동하는 시장은 매우 개방된 환경을 만든다. 이는 그러한 앱들 중 하나가 악의적인 의도를 가장하고 디바이스 상의 다른 앱들을 훼손하거나 이것들로부터 절도하기 시작할 때까지는 매우 사용자 우호적이다. 디바이스가 예전대로와 동일한 디바이스인지 여부를 인지하는 것에 더하여, 서비스 제공자는 디바이스가 예전과 동일한 상태에 있는지를 디바이스에 물어봐야 한다. 상당한 변화들이 일어났던 것으로 알려질 때, 이는 잠재적 위협을 나타낼 수 있다. 이러한 인지는 서비스 제공자들이 개선책을 취하거나 기계가 여전히 안전하다는 디바이스 작동자로부터의 추가 확인을 적어도 요청하는 것을 가능하게 한다.

[0009] 사용자는 사용자의 디바이스가 위해되는지 여부를 흔히 인지하지 않을 것이지만, 예를 들어, 바이오스가 변경되었다는 것이 검출될 수 있으면, 서비스는 경고 단계들을 취할 수 있다.

[0010] 앱들을 설치하고 실행하는 것은 매우 단순한 것으로 여겨진다. 그러나, 앱들의 근원의 강한 보장 및 다른 앱들의 실행으로부터의 불투명한 구분으로부터 크게 이익을 얻을 수 있는 부류의 앱들이 있다. 이는 예를 들어, 신뢰된 실행 환경 또는 TEE일 수 있다. 주 OS 및 메모리 스택 상에서 실행되는 앱과는 달리, TEE에서 실행되는 앱은 OS에 의한 스누핑 없이 발휘될 수 있는 암호화 프리미티브들에 대한 액세스를 가질 수 있다. 이상적 상황들에서, TEE에서 실행되는 앱은 디바이스의 작동자와의 사적 상호 작용을 보장하기 위해 사용자 입력 및 디스플레이에 대한 직접적 액세스를 또한 갖는다.

[0011] 디바이스 보안의 지원에서의 전매 및 표준 기반 솔루션들 둘 다는 공급 체인으로의 솔루션들 둘 다의 방식으로 나아가 왔다. 신뢰된 플랫폼 모듈 또는 TPM은 예를 들어, 대부분의 현대 PC들의 마더보드 상에 내장되는 보안 칩이다. 상기 기술은 수십개의 주요 판매 회사의 비영리 컨소시엄인, 신뢰된 컴퓨팅 그룹(TCG)에 의해 지정된다. 상기 기술은 대체로 기업 네트워크 보안의 지원으로 설계되었지만 소비자 웹을 단순화하는데 막대한 역할을 갖는다. TPM들은 6 년 동안 출하되고 있었고 이제 현대 PC들에서 광범위하게 널리 퍼져 있다. 2015 년에서 시작한 Microsoft 로고 준수는 어떤 기계도 TPM 없이 납품되지 않는다는 것을 추가로 보장한다.

[0012] TPM은 비교적 단순하다. TPM은 3가지 기본 목적: PKI, 바이오스 무결성 및 암호화를 제공한다. 기술이 10 년이 훨씬 넘는 동안 계속되었지만, TEE에 대한 지원을 갖는 디바이스들이 이용 가능해졌던 것은 단지 최근이다. Intel은 2011 년에 상업 솔루션들의 납품을 시작하였고 Trustonic이 2013 년에 런칭되었다. 플랫폼들 및 연관된 툴들은 소비자 사용에 필요한 원숙의 레벨에 도달하고 있다. TEE로 앱을 전개시키는 것은 전용 하드웨어 디바이스를 납품하는 것과 유사하다. 실행 및 데이터는 호스트의 임의의 다른 기능으로부터 암호학적으로 격리된다.

[0013] 칩은 칩 자체의 어떤 신원도 갖지 않지만, 키 쌍들을 생성하도록 요청될 수 있다. AIK들 또는 입증 신원 키들은 키 쌍의 개인 절반이 하드웨어 외부에서 결코 알아 볼 수 없도록 "이동할 수 없는" 것으로 표시될 수 있다. 이는 복제될 수 없는 기계 신원을 확립할 기회를 제공한다. 현재 활용되는 TPM들인, 버전 1.2는 RSA 및 SHA-1에 제한된다. 곧 나오는 버전 2.0은 훨씬 더 기민할 것이다. TPM은 또한 보증 키(EK)를 구현한다. EK는 제조하는 동안 설치되고 TPM이 실제로 진품의 TPM이라는 것을 입증하는데 사용될 수 있다. TPM을 지원하는 시스템은 TPM을 지원하는 시스템의 부트 시퀀스 동안 플랫폼 구성 레지스터들(PCR's)을 로딩할 것이다. 펌웨어로 시작하여, 부트 프로세스의 각각의 단계는 부트 프로세스의 상태 및 다음 프로세스의 상태를 측정하고 PCR 값을 기록한다. PCR들이 부정 조작이 안되는 TPM에서 캡처됨에 따라, 시스템의 바이오스 무결성의 신뢰 가능한 "인용구"가 이후에 요청될 수 있다. PCR은 무엇이 실제로 일어났는지를 캡처하지 않고, 어떤 것도 변경되지 않는다는 것만을 일련의 해시들을 통해 캡처한다. 이는 가장 심각한 것 그리고 그렇지 않으면 해커가 기계 바이오스를 위해시키거나 기밀 하이퍼바이저를 설치하는 검출할 수 없는 공격들에 대한 보호에 특히 중요하다. 바이러스 스캐닝 소프트웨어로부터의 보장 시그니처와 결합되므로, 기계 헬스의 신뢰 가능한 상태를 확립할 수 있다. TPM들은 또한 벌크 암호화 서비스들을 제공한다. 암호화 키들은 TPM에서 생성되지만, TPM에 저장되지 않는다. 대신에, 암호화 키들은 TPM 결속 저장 루트 키로 암호화되고 요청 프로세스로 복귀된다. 데이터의 블랍을 암호화하거나 복호화하기를 원하는 프로세스는 우선 원하는 키를 장착할 것이다. 키는 그 다음 하드웨어에서 복호화되고 암호화하는데 이용 가능해진다. 대부분의 TPM 키로와 같이, 암호화 키들은 원한다면 비밀 번호로 추가도 보호될 수 있다.

[0014] Trustonic(http://www.trustonic.com)은 ARM, G+D 및 Gemalto의 합작 기업이다. Trustonic은 광범위한 스마트 디바이스에 걸쳐 신뢰된 실행 환경을 제공한다. 목표는 민감한 애플리케이션 서비스들의 보안적 실행을 가능하게 하는 것이다. Trustonic은 신뢰된 실행 환경들에 대한 범용 플랫폼 표준의 구현이다. Trustonic TEE에서 실행시키기 위해 기록되는 앱들이 서명되고 측정된다. Trustonic을 지원하는 디바이스들은 루팅된 디바이스 상의 디버그 작동들을 포함하여, 로딩된 앱이 디바이스 상에서 실행되는 임의의 다른 프로세스에 의해 감시될 수 없도록 격리된 실행 커널을 제공한다. Trustonic은 2012 년에 성립되었고 이제 6개의 제조품을 출하하고 24개의 서비스 제공자들을 지원한다. 2억개가 넘는 디바이스가 지금 Trustonic 지원으로 출하되었다.

[0015] Intel vPro는 현대 Intel 칩 세트로 구축되는 기술들의 컬렉션이다. vPro와 함께 판매되는 새로운 기계들은 TXT 신뢰된 실행 기술을 지원한다. Intel은 많은 암호화 기능의 보호된 실행을 가능하게 하는 관리 엔진(ME)에서의 보안적 처리 환경을 제공한다. 이러한 능력의 한가지 사용은 ME에서의 앱으로서 구현되는 TPM 2.0 기능성의 활용이었다. 관리 엔진은 또한 사용자와의 완전히 격리된 통신을 행하는 보안적 디스플레이 기능들을 지원한다. 이러한 방식으로, ME에서 실행되는 앱은 실질적으로 감소된 위해의 위험성으로 사용자로부터의 지시를 취할 수 있다.

[0016] ARM 트러스트존은 모든 ARM 프로세서 상에서 이용 가능한 실리콘 기반들을 제공한다. 프리미티브들은 통상의 실행 공간으로부터 실행의 보안된 세계를 격리시킨다. ARM은 그 다음 다수의 표준 프로세서로 구축되는 설계들을 제공한다. 트러스트존을 이용하기 위해, 앱들은 제조자에 의한 시스템 펌웨어의 일부로서 활용될 수 있거나 Trustonic, Linaro 또는 Nvidia의 개방 소스 마이크로 커널 같은 제3 자 툴들을 통해 사후에 전해질 수 있다.

[0017] 본 발명의 일부 실시예는 사람들 및 블록 체인을 연결시키는 트랜잭션 환경을 강화시키는 서비스들의 세트로 이러한 기술들을 적용시킨다.

[0018] 제2 요소 승인의 개념은 제한된 사용에서이지만 양호하게 확립된다. 제2 요소 승인의 개념은 아마도 로그인을 뚫는 것이 자금들의 즉각적이고 철회할 수 없는 절도를 제공할 수 있는 비트코인 서비스 사이트들에 의해 가장 두드러지게 활용된다. 대부분의 사람들은 SMS 확인 또는 전자 열쇠의 형태의 제2 요소에 친숙하다. 이들은 사용자 이름 및 비밀 번호를 입력하고 그 다음 등록된 전화기로 메시징되는 코드를 입력한다. 제2 요소 승인은 로그인 보안에 대해 중요한 단계이지만, 사용자에게 부가 작업을 부담시킨다. 왜 이것이 중요한지를 이해하더라도, 사람들은 본래 나태하다. 많은 사이트는 사용자들이 반복된 확인들을 거부하여 많은 사용자가 이러한 시간 절약하는 것을 손쉽게 선택하여 보안을 저하시킨다. 추가 예시적 방법은 승인 요청이 송신되는 디바이스로 우선 입증하게 될 수 있다. TPM 또는 암호화 키 세트들의 임의의 다른 보안적 소스를 사용하여, 웹 서비스는 디바이스가 예전과 동일한 디바이스라는 것을 입증할 것을 디바이스에 요청할 수 있다. 이러한 요청은 사용자에게 투명할 (또는 핀으로 추가로 보안될) 수 있고 사용자가 싫어하는 신원 및 승인에 대해 흔히 우회될 수 있는 보장의 레벨을 제공한다.

[0019] 기계 생성 암호화 입증은 둘 다가 아마도 사용자에 기인하는 기억할 만한 사실들에 기반하는 짧은 사용자 이름 및 8개의 문자 비밀 번호보다 훨씬 더 신뢰 가능한 경향이 있다. 사용자는 디바이스를 보호하는 일로 가장 적합된다. 수만년의 진화는 값비싼 대상들을 보호하도록 사람들을 훈련시켰다. 그러나 사람은 10자리수 전화 번호도 기억하기 어렵다는 것을 알 수 있다. 다른 한편으로는, 디바이스들은 뚜렷하게 빠른 계산을 위해 특별히 만들어진다. 사용자가 정기적으로 사용되는 디바이스 없이 사용자 자신이 찾는 경우, 서비스는 사용자 신원 확인 절차들에 의지할 수 있다. 그것이 통상의 사용 사례가 아닐 때, 사용자는 더 번거로운 신원 확인 절차들을 기꺼이 수락할 것이다.

[0020] 본 발명의 예시적 실시예에 따르면, 디바이스 신원을 레버리징하는 제1 단계는 등록이다. 하나의 바람직한 실시예에서, 디바이스 등록은 일부 다른 신뢰된 엔티티의 감시 하에서 행해질 수 있다. 예를 들어, 전화기의 등록은 최종 사용자와 디바이스 신원 사이의 결속이 물리적 존재로 확립될 수 있는 판매하려는 순간에 일어날 수 있다. 그러나 많은 사용 사례에서, 이러한 레벨의 사람 대 디바이스 연관성은 필요하지도 않고 원해지지도 않는다. 개인 식별 정보(PII)로서 고려될 수 있는 디바이스 신원 및 속성들은 불가분하게 링크되지 않아야 한다. 기본 디바이스 신원은 순전히 익명이다. 디바이스를 신뢰할 수 있게 등록하기 위해, 단지 2개의 것: A) 디바이스로 락킹되는 키 쌍을 생성하는 능력, 및 B) 이러한 서비스를 제공하는 디바이스 환경의 출처 및 품질의 보장을 필요로 한다. 후자는 사회 공학 또는 공급 체인 암호화에 의해 제공된다. 어떤 것도 절대적이지 않지만, 평판 있는 납품 업자의 존재에서 등록되는 디바이스는 실제의 디바이스일 가능성이 있다. 이는 그러한 납품 업자의 지속하는 평판에 중요하다. 제조 작업장에 초점이 맞춰지고 OEM 인증 기관으로 확인될 수 있는 디바이스의 신뢰는 마찬가지로, 제조자의 평판에 의거한다.

[0021] 일부 실시예들에 따르면, 등록은 조회되지만 도용되지 않을 수 있는 고유성을 확립하는 것을 수반한다. 이를 위해, TPM (또는 유사한 신뢰의 하드웨어 루트)가 사용될 수 있다. TPM 칩은 키 쌍을 생성하고 결국 서버로 키의 공적 부분을 발송하는 클라이언트로 키의 공적 부분을 복귀시킨다. 랜덤 id가 생성되고 함께 쿠플렛(couplet)이 네임코인 (또는 명명된 데이터를 기록하도록 고안되는 유사한 블록 체인, 또는 블록 체인 방법)으로 트랜잭션된다. 블록 체인에 안치되면, 디바이스 기록은 연장되고 PCR 인용구들, 연관된 비트코인 계정들 또는 다른 데이터와 같은 속성들로 변경될 수 있다. 큰 데이터 객체들이 직접보다는 오히려 블록 체인에서의 해시 및 URL로 참조될 것이라는 것이 예상된다. 등록 에이전트는 디바이스와 함께, 이러한 기록을 업데이트할 수 있는 네임코인 계정을 제어한다. 그러나, 등록 에이전트가 또한 디바이스인 자체 등록된 디바이스들에 대한 시나리오를 가정할 수 있다. 등록되면, 서비스는 연관된 속성들이 디바이스에서 나온다는 통신 및 암호화 보장을 입증하고 암호화하기 위해 디바이스의 공개 키들에 액세스할 수 있다.

[0022] 신뢰된 실행 환경에서, 디바이스 신원의 특징부들은 시스템의 나머지로부터 별개로 코드를 실행시키는 능력을 추가로 연장하면서, 제공된다. 본 발명의 실시예들은 다양한 TEE 환경에서의 활용을 위해 패키징되는 비트코인 서비스 구성 요소를 제공한다. 이는 이하의 트랜잭션의 실행에 대한 수개의 중대한 강화를 야기한다: (1) 코드

는 조작될 수 없도록 제3 자 신뢰된 애플리케이션 관리자에 의해 서명되고 승인된다. (2) 코드는 호스트 작동 환경 외부에서 실행되고 따라서 멀웨어로부터 보호된다. (3) 단지 키들 너머의 애플리케이션 데이터는 TEE의 외부에 결코 노출되지 않는다.

[0023] 등록된 디바이스는 서비스 제공자들이 등록된 디바이스의 상태 및 콘텍스트를 검증하는 것을 가능하게 하는 속성들의 기록을 축적할 수 있다. 디바이스 속성들은 유용할 임의의 PII를 포함할 필요가 없다. 예를 들어, 클린 부트 시퀀스를 언명하는 최근의 진술은 기계가 위해되지 않는다는 일부 확신을 서비스 제공자에게 부여할 수 있다. 사실의 개개 주장을 제공하는 속성들은 많은 PII를 누설하지 않고 유용할 수도 있으며, 예를 들어, 기계 작동자는 21 살을 넘는 것으로, 또는 프랑스 시민 또는 동호 클럽의 구성원으로 입증되었다. 대부분의 경우에, 디바이스와의 상호 작용은 디바이스의 부트 무결성의 진술을 수집할 기회이다. 이는 지난 부트 진술에 대하여 비교될 수 있는 해시들의 컬렉션에 불과하다. 예측 가능한 방식으로 부팅하였던 기계는 바이오스 또는 OS를 변경했던 자보다 믿을 수 있게 더 신뢰 가능하다. PCR 인용구들에 더하여, 참여하는 안티바이러스 소프트웨어는 기계가 마지막 스캔 시점에 삭제되었다는 진술을 전할 수 있다.

[0024] 일부 실시예들에서, 신뢰된 네트워크 연결(TNC)의 원리들의 통합은 트랜잭션의 수락 이전에 알려지지 않은 클라이언트 디바이스의 전체 입증을 가능하게 할 것이다. 알려진 양호한 조건 또는 트랜잭션의 수락 이전의 상태로 있는 클라이언트 디바이스는 디바이스가 정확하게 구성된다는 제3 자의 진술에 기반한다. 이러한 타입의 검증은 바람직하게는 임의의 트랜잭션 처리 시스템의 일부로서 필요할 수 있는 광범위한 사이버 보안 제어를 다룬다.

과제의 해결 수단

[0025] 예시적인 실시예는 블록 체인 네트워크에서의 전자 트랜잭션을 전달하는 것의 대비로, 트랜잭션의 일부로서 디바이스 무결성 검증 프로세스를 구현하는 단계로서, 사용자 디바이스에서의 신뢰의 루트로부터 디바이스 실행 환경의 무결성의 내부 입증을 수행하는 단계; 및 전자 시그니처를 필요로 하여, 시그니처의 무결성의 검증이 블록 체인 트랜잭션에 적용되는 단계로서; 시그니처의 무결성의 검증은 디바이스의 실행 환경이 알려진 양호한 조건으로 있는지 여부의 판단에 기반하며: 시그니처의 무결성에 기반하여, 트랜잭션이 진행되는 것을 가능하게 하거나, 디바이스의 실행 환경이 알려진 양호한 조건으로 있지 않다는 것이 판단되더라도 사용자에 의해 의도되는 바에 따라 전자 트랜잭션이 진행되는 것이 가능해지는 것을 검증하기 위해 개선 권한을 요청하는 단계를 포함하는 단계를 포함하는 단계를 포함하는 블록 체인 통신 네트워크에서의 사용자 디바이스의 디바이스 무결성을 검증하는 컴퓨터 구현 방법이다. 일부 실시예들에서, 시그니처의 무결성의 검증은 처리를 위해 블록 체인 네트워크에 신뢰 명령어의 루트를 송신하여, 블록 체인 네트워크의 적어도 일부가 전자 트랜잭션을 수락하기 위해 다수의 전자 시그니처를 필요로 함으로써 응답하는 단계로서, 디바이스의 실행 환경 내에서, 사용자 디바이스에서의 신뢰의 루트로부터 명령어를 생성하는 단계; 신뢰 명령어의 루트에 상응하는 제1 전자 시그니처를 필요로 하여, 시그니처의 무결성의 검증이 블록 체인 트랜잭션에 적용되는 단계; 및 디바이스의 실행 환경이 알려진 양호한 조건으로 있는지 여부의 판단에 기반하여 시그니처의 무결성을 검증함으로써 제1 전자 시그니처에 응답하는 단계로서, 시그니처를 앞서 기록된 참조값과 비교하는 단계; 시그니처가 앞서 기록된 참조값에 부합하면, 그 때 트랜잭션이 진행되는 것을 가능하게 하는 단계; 및 시그니처가 앞서 기록된 참조값에 부합하지 않으면, 디바이스의 실행 환경이 알려진 양호한 조건으로 있지 않다는 것이 판단되더라도 사용자에 의해 의도되는 바에 따라 전자 트랜잭션이 진행되는 것이 가능해지는 것을 검증하기 위해 제3 자 대역 외 프로세스를 요청하는 단계를 포함하는 단계를 포함하는 단계를 포함한다. 일부 실시예들에서, 시그니처의 무결성을 검증하는 단계는 디바이스의 실행 환경이 알려진 양호한 조건으로 있는지 여부의 판단에 기반하여 디바이스가 전자 시그니처를 제공하는 단계; 디바이스가 전자 시그니처를 제공하면, 트랜잭션이 진행되는 것을 가능하게 하는 단계; 개선 권한이 시그니처를 제공하면, 디바이스의 실행 환경이 알려진 양호한 조건으로 있지 않다는 것이 판단되더라도 사용자에 의해 의도되는 바에 따라 트랜잭션이 진행되는 것을 가능하게 하는 단계를 포함한다. 게다가, 대역 외 프로세스는 사용자의 의도가 미리 결정된 필요 조건들을 충족시키거나, 디바이스 무결성이 미리 결정된 필요 조건들을 충족시키거나, 부가 프로세스가 미리 결정된 필요 조건들을 충족시킨다는 것 중 적어도 하나를 확인하기 위해 N 또는 M 암호화 키 기능을 사용하는 단계를 더 포함할 수 있다. 참조값은 디바이스 플랫폼의 소유주에 의해 수행되는 등록 프로세스 동안 생성될 수 있다. 참조값은 디바이스에 할당되는 버스(birth) 증명서에 기반하여 생성될 수 있으며, 버스 증명서는 디바이스의 제조자 또는 생성자, 디바이스의 실행 환경의 제조자 또는 생성자, 및/또는 디바이스 상의 애플리케이션의 제조자 또는 생성자에 의해 생성된다. 참조값은 디바이스의 제조자 또는 생성자, 디바이스의 실행 환경의 제조자 또는 생성자, 및/또는 디바이스 상의 애플리케이션의 제조자 또는 생성자 중 적어도 하나의 시그니처를 포함할 수 있다. 제3 자 대역 외 프로세스는 트랜잭션을 검증하기 위해 요청에 응하여 토큰을 복귀시킬 수 있다. 일부 실시예는 시그니처가 앞서 기록된 참조값에 부합하지 않으면, 전자 트랜잭

션이 일정 기간 내에 완료되는 것을 가능하게 할 수 있다.

[0026] 일부 실시예는 참조값의 등록과 트랜잭션 사이의 기간 및/또는 트랜잭션의 양에 기반하여 디바이스의 실행 환경이 알려진 양호한 조건으로 있지 않다는 것이 판단되더라도 의도된 전자 트랜잭션이 진행되는 것이 가능해지는 것을 검증할 수 있다. 임계량을 넘는 트랜잭션들은 기간이 미리 결정된 필요 조건들을 충족시키면, 진행되는 것이 가능해질 수 있다. 일정량을 넘는 트랜잭션을 가능하게 하는 것은 최소 수의 앞서 가능해진 트랜잭션에 기반할 수 있다. 일부 실시예는 디바이스 무결성이 최소 미리 결정된 필요 조건을 충족시키는지 여부 및 취해질 추가 작동들을 사용자에게 나타내는 디스플레이 디바이스를 사용하는 단계를 더 포함할 수 있다. 다른 실시예들은 제3 자에게의 트랜잭션의 통지를 더 포함할 수 있으며, 통지에 응하여, 제3 자는 트랜잭션 및 디바이스의 상태를 기록한다. 제3 자는 트랜잭션의 장래 분석을 위해 디바이스 무결성과 연관된 측정치들을 기록할 수 있다. 게다가, 기록의 프라이버시를 보장하는 것은 기록이 인증된 제3 당사자들에게만 이용 가능해지도록 기록을 암호로 혼란하게 하는 것을 포함할 수 있다. 다른 예시적인 실시예는 블록 체인 통신 네트워크; 블록 체인 네트워크에서의 사용자 디바이스; 블록 체인 네트워크에서의 전자 트랜잭션; 블록 체인 네트워크에서의 전자 트랜잭션의 전달의 대비로 트랜잭션의 일부로서 구현되는 디바이스 검증 프로세스를 포함하는 블록 체인 통신 네트워크에서의 사용자 디바이스의 디바이스 무결성을 검증하는 컴퓨터 구현 시스템이며, 구현은 디바이스에서의 신뢰의 루트로부터 수행되는 디바이스 실행 환경의 무결성의 내부 입증; 시그니처의 무결성의 검증이 블록 체인 트랜잭션에 적용되기 위한 전자 시그니처로서; 시그니처의 무결성의 검증은 디바이스의 실행 환경이 알려진 양호한 조건으로 있는지 여부의 판단에 기반하며: 시그니처의 무결성에 기반하여, 트랜잭션이 진행되는 것을 가능하게 하거나, 디바이스의 실행 환경이 알려진 양호한 조건으로 있지 않다는 것이 판단되더라도 사용자에 의해 의도되는 바에 따라 전자 트랜잭션이 진행되는 것이 가능해지는 것을 검증하기 위해 개선 권한을 요청하는 것을 포함하는 전자 시그니처를 더 포함한다.

도면의 간단한 설명

[0027] 전술한 것은 참조 문자들이 상이한 도면들 전체에 걸쳐 동일한 부분들을 지칭하는 첨부 도면들에 도시되는 바와 같은 본 발명의 예시적 실시예들의 이하의 보다 특정한 설명으로부터 명백할 것이다. 도면들은 반드시 일정 비율로 그려지는 것은 아니고, 강조가 대신에 본 발명의 실시예들을 예시할 시에 배치된다.

도 1a는 본 발명의 실시예들이 구현될 수 있는 예시적 디지털 처리 환경이다.

도 1b는 컴퓨터/컴퓨팅 노드의 임의의 내부 구조체의 블록도이다.

도 2a는 본 발명에 따른 예시적 디바이스 승인 시스템을 도시하는 블록도이다.

도 2b는 본 발명에 따른 예시적 디바이스 승인 시스템을 도시하는 도면이다.

도 2c는 본 발명의 일 실시예의 구성 요소들의 도면이다.

도 2d는 승인 시스템 어댑터 및 승인 시스템 어댑터의 밖으로 그리고 안으로 본 인터페이스들의 도면이다.

도 3a는 인코더에 의해 명령어를 패키징하고 전달하는 시퀀스의 도면이다.

도 3b는 본 발명의 일 실시예에 따른 디바이스 등록 프로세스의 도면이다.

발명을 실시하기 위한 구체적인 내용

[0028] 본 발명의 예시적 실시예들의 설명이 뒤따른다.

[0029] 본 발명의 실시예들은 전자 트랜잭션들에 관여하기 이전에 디바이스 헬스를 입증하는 시스템들 및 방법들이다.

[0030] 블록 체인 트랜잭션들은 트랜잭션들을 수행하는 알려지지 않은 디바이스 상의 검증 또는 사이버 보안 제어들을 갖지 않는다. 그러므로, 블록 체인 트랜잭션의 수락 이전에 알려지지 않은 클라이언트 디바이스의 전체 입증은 블록 체인 트랜잭션들에 대한 보안을 추가로 제공할 것이다.

[0031] 예시적 실시예들은 디바이스의 무결성이 네트워크 스위치에의 연결의 실제 활성화 이전에 검증될 수 있는 신뢰된 네트워크 연결(TNC) 표준들의 원리들에서 볼 수 있다. TNC에 따르면, 디바이스는 디바이스 상에 보안적으로 저장되는 일련의 측정들을 수행한다. 측정들은 전형적으로 바이오스 이미지, 운영 체제(OS) 및 임의의 애플리케이션들이 바뀌어지지 않았다는 것이 검증될 필요가 있는 바이오스 이미지, 운영 체제(OS) 및 임의의 애플리케이션들의 입증을 포함할 것이다. 네트워크에의 연결 시에, 스위치는 디바이스가 앞서 연결되었거나 현재의 알려진

양호한 조건 또는 상태에 있을 때, 컴퓨팅되었던 참조값에 측정 데이터가 부합한다는 것을 검증하는 입증 프로세스를 수행할 것이다. 신뢰된 실행 환경(TEE)은 또한 디바이스의 자체 측정 프로세스들 및 헬스의 원격 입증이 가능하다. 일부 바람직한 실시예들에서, TNC 시스템은 신뢰된 컴퓨팅 그룹(TCG) 표준들에 기반하고 전형적으로 신뢰된 플랫폼 모듈(TPM) 칩은 통합된다.

[0032] 일부 실시예들에서, 전체 디바이스 무결성 검증의 자동화가 블록 체인 트랜잭션의 일부로서 제공된다. 디바이스 무결성의 입증을 제공하기 위해, 블록 체인 명령어를 수행하는 디바이스는 블록 체인 트랜잭션의 초기화에서 디바이스의 신뢰의 루트로부터 실행 환경의 무결성의 내부 입증을 수행할 것이다. 디바이스는 사람 입력으로 또는 이것 없이, 측정된 환경 내에서 명령어를 생성할 것이다. 이러한 명령어는 그 다음 처리를 위해 블록 체인 네트워크로 송신될 것이다. 블록 체인 네트워크는 트랜잭션을 수락하는데 다수의 시그니처를 필요로 할 것이다. 제1 시그니처는 트랜잭션에 적용되는 시그니처의 검증을 가질 생성된 루트 명령어 그 자체일 것이다. 네트워크는 그 다음 제1 시그니처를 앞서 기록된 참조값과 비교함으로써 실행 환경의 무결성 시그니처를 검증한다. 시그니처가 참조값에 부합하면, 트랜잭션은 진행되는 것이 가능해진다. 시그니처 및 참조값이 부합하지 않으면, 그 때 시스템은 실행 환경이 알려진 양호한 조건에 있지 않더라도 의도되는 트랜잭션이 진행되는 것이 가능해진다는 것을 검증할 제3 대역 외 프로세스가 완료되는 것을 필요로 할 것이다. 블록 체인 트랜잭션들이 트랜잭션을 수행하는 알려지지 않은 디바이스 상의 임의의 검증 또는 사이버 보안 제어들을 갖지 않으므로, 본 발명의 실시예들은 디바이스가 트랜잭션의 수락 이전에 정확하게 구성되었다는 제3 자의 진술에 따라 알려지지 않은 클라이언트 디바이스가 알려진 양호한 조건으로 있는 전체 입증을 가능하게 할 것이다. 그러므로, 본 발명의 일부 실시예는 임의의 블록 체인 트랜잭션 처리 시스템의 일부로서 필요할 광범위한 사이버 보안 제어를 다룰 수 있다.

[0033] 디지털 처리 환경

[0034] 트랜잭션들에 관여하기 이전에 디바이스 헬스를 입증하는 본 발명에 따른 시스템(100)의 예시적 구현은 소프트웨어, 펌웨어 또는 하드웨어 환경으로 구현될 수 있다. 도 1a는 본 발명의 실시예들이 구현될 수 있는 하나의 그러한 예시적 디지털 처리 환경을 도시한다. 클라이언트 컴퓨터들/디바이스들(150) 및 서버 컴퓨터들/디바이스들(160) (또는 클라우드 네트워크(170))는 애플리케이션 프로그램들 등을 실행시키는 처리, 저장 및 입력/출력 디바이스들을 제공한다.

[0035] 클라이언트 컴퓨터들/디바이스들(150)은 다른 클라이언트 컴퓨터들/디바이스들(150) 및 서버 컴퓨터/디바이스들(160)을 포함하는 다른 컴퓨팅 디바이스들로 직접 또는 통신 네트워크(170)를 통해 링크될 수 있다. 통신 네트워크(170)는 무선 또는 유선 네트워크, 원격 액세스 네트워크, 범용 네트워크(즉, 인터넷), 컴퓨터들의 전 세계적인 컬렉션, 로컬 영역 또는 광역 네트워크들, 및 서로와 통신하기 위해 현재 다양한 프로토콜을 사용하는 게이트웨이들, 라우터들 및 스위치들(예를 들어 TCP/IP, Bluetooth®, RTM 등)의 일부일 수 있다. 통신 네트워크(170)는 가상 사설 네트워크(VPN) 또는 대역 외 네트워크 또는 둘 다일 수도 있다. 통신 네트워크(170)는 데이터 네트워크, 음성 네트워크(예를 들어 지상 통신선, 모바일 등), 오디오 네트워크, 영상 네트워크, 위성 네트워크, 무선 네트워크 및 페이저 네트워크를 포함하지만, 이에 제한되지 않는 다양한 형태를 취할 수 있다. 다른 전자 디바이스/컴퓨터 네트워크 아키텍처들이 또한 적절하다.

[0036] 서버 컴퓨터들(160)은 승인 시스템에 의해 보호되는 리소스들에 요청자가 액세스하는 것을 가능하게 하기 이전에 요청자의 신원을 확인하도록 승인자들과 통신하는 사용자 디바이스 승인 시스템(100)을 제공하도록 구성될 수 있다. 서버 컴퓨터들은 별도의 서버 컴퓨터들이 아니고 클라우드 네트워크(170)의 일부일 수 있다.

[0037] 도 1b는 오디오, 이미지, 영상 또는 데이터 신호 정보를 표시하는 것을 용이하게 하는데 사용될 수 있는 도 1a의 처리 환경에서의 컴퓨터/컴퓨팅 노드(예를 들어, 클라이언트 프로세서/ 디바이스(150) 또는 서버 컴퓨터들(160)) 임의의 내부 구조체의 블록도이다. 도 1b의 각각의 컴퓨터(150, 160)는 시스템 버스(110)를 포함하며, 버스는 컴퓨터 또는 처리 시스템의 구성 요소들 중의 데이터 전송을 위해 사용되는 실제 또는 가상 하드웨어 라인들의 세트이다. 시스템 버스(110)는 본질적으로 요소들 사이의 데이터의 전송을 가능하게 하는 컴퓨터 시스템의 상이한 요소들(예를 들어, 프로세서, 디스크 저장, 메모리, 입력/출력 포트들 등)을 연결시키는 공유된 전선관이다.

[0038] 다양한 입력 및 출력 디바이스(예를 들어, 키보드, 마우스, 터치 스크린 인터페이스, 디스플레이, 프린터, 스피커, 오디오 입력 및 출력, 영상 입력 및 출력, 마이크 잭 등)를 컴퓨터(150, 160)에 연결시키는 I/O 디바이스 인터페이스(111)가 시스템 버스(110)에 부착된다. 네트워크 인터페이스(113)는 네트워크(예를 들어, 도 1a의 170으로 도시된 네트워크)에 부착되는 다양한 다른 디바이스에 컴퓨터가 연결되는 것을 가능하게 한다. 메모리(114)는 본 발명의 일부 실시예의 디바이스 무결성 입증 및 승인 구성 요소들의 소프트웨어 구현들을 구현하는

데 사용되는 컴퓨터 소프트웨어 명령어들(115) 및 데이터(116)에 대한 휘발성 저장소를 제공한다. 본원에 설명하는 사용자 승인 시스템(100)의 그러한 디바이스 무결성 입증 및 승인 소프트웨어 구성 요소들(115, 116)(예를 들어, 도 2a의 인코더(210), 신뢰된 실행 환경(TEE) 애플릿(208), 승인 사이트(206))은 Python과 같은 임의의 높은 레벨의 객체 지향 프로그래밍 언어를 포함하는 임의의 프로그래밍 언어를 사용하여 구성될 수 있다.

[0039] 예시적 모바일 구현에서, 본 발명의 모바일 에이전트 구현이 제공될 수 있다. 클라이언트 서버 환경은 서버(190)를 사용하여 모바일 보안 서비스들을 가능하게 하는데 사용될 수 있다. 클라이언트 서버 환경은 디바이스(150) 상의 디바이스 승인 엔진/에이전트(115)를 서버(160)에 테더링(tethering)하기 위해 예를 들어, XMPP 프로토콜을 사용할 수 있다. 서버(160)는 그 다음 요청 시에 모바일 전화기로 커맨드들을 발할 수 있다. 시스템(100)의 일정 구성 요소들에 액세스하는 모바일 사용자 인터페이스 프레임워크는 XHP, Javelin 및 WURFL에 기반할 수 있다. OS X 및 iOS 운영 체제들 및 OS X 및 iOS 운영 체제들의 각각의 API에 대한 다른 예시적 모바일 구현에서, Cocoa and Cocoa Touch는 C 프로그래밍 언어에 작은 대화 스타일 메시징을 추가하는 오브젝티브 C 또는 임의의 다른 높은 레벨 프로그래밍 언어를 사용하여 클라이언트측 구성 요소들(115)을 구현하는데 사용될 수 있다.

[0040] 시스템은 사용자를 등록하고, 요청자가 등록된 사용자라는 것을 확인하기 위해 승인자들/입증자들을 선택하고, 요청자의 신원을 확인하는 것에 대하여 승인들과 통신하고, 시스템에 의해 보호되는 리소스들에 대한 요청자 액세스를 허용하거나 거부하도록 신뢰 점수들을 컴퓨팅하기 위해 통계적 알고리즘들과 같은 알고리즘들을 실행시키는 것을 가능하게 하는 승인 (또는 입증) 엔진(240)(도 2)을 포함할 수 있는 서버 컴퓨터들(160) 상의 서버 프로세스들의 인스턴스들을 포함할 수도 있다.

[0041] 디스크 저장소(117)는 시스템(100)의 실시예들을 구현하는데 사용되는 컴퓨터 소프트웨어 명령어들(115)(동등하게 "OS 프로그램") 및 데이터(116)에 대한 비휘발성 저장소를 제공한다. 시스템은 서버 컴퓨터(160)에 액세스 가능한 디스크 저장소를 포함할 수 있다. 서버 컴퓨터는 시스템(100)에 등록되는 사용자들의 승인과 관련되는 기록들에 대한 보안적 액세스를 유지 관리할 수 있다. 중앙 처리 장치(112)는 또한 시스템 버스(110)에 부착되고 컴퓨터 명령어들의 실행을 제공한다.

[0042] 예시적 실시예에서, 프로세서 루틴들(115) 및 데이터(116)는 컴퓨터 프로그램 제품들이다. 예를 들어, 승인 시스템(100)의 양태들은 서버측 및 클라이언트측 구성 요소들 둘 다를 포함할 수 있다.

[0043] 예시적 실시예에서, 승인자들/입증자들은 모두가 소프트웨어(115, 116)로 적어도 부분적으로 구현될 수 있는 인스턴트 메시징 애플리케이션들, 화상 회의 시스템들, VOIP 시스템들, 이메일 시스템들 등을 통하여 연락될 수 있다. 다른 예시적 실시예에서, 승인 엔진/에이전트는 컴퓨팅 디바이스(150) 상에서 실행되는 신뢰된 플랫폼 모듈(TPM) 상의 사용자들을 승인하도록 구성되는 애플리케이션 프로그램 인터페이스(API), 실행 가능 소프트웨어 구성 요소 또는 OS의 통합된 구성 요소로서 구현될 수 있다.

[0044] 소프트웨어 구현들(115, 116)은 사용자 승인 시스템(100)에 대한 소프트웨어 명령어들의 적어도 일부를 제공하는 저장 디바이스(117) 상에 저장될 수 있는 컴퓨터 판독 가능 매체로서 구현될 수 있다. 승인 엔진의 인스턴스들과 같은 사용자 승인 시스템(100)의 각각의 소프트웨어 구성 요소의 인스턴스들을 실행시키는 것은 컴퓨터 프로그램 제품들(115)로서 구현될 수 있고, 관련 분야에 널리 알려져 있는 바와 같은 임의의 적절한 소프트웨어 설치 절차에 의해 설치될 수 있다. 다른 실시예에서, 시스템 소프트웨어 명령어들(115) 중 적어도 일부는 (모바일로부터 실행되든 아니면 다른 컴퓨팅 디바이스로부터 실행되든) 예를 들어, 브라우저 SSL 세션을 통하여 또는 앱을 통해 케이블, 통신 및/또는 무선 연결을 통하여 다운로드될 수 있다. 다른 실시예들에서, 시스템(100) 소프트웨어 구성 요소들(115)은 전파 매체(예를 들어, 전파, 적외선파, 레이저파, 음향파, 또는 인터넷 또는 다른 네트워크들과 같은 범용 네트워크를 통해 전파되는 전기파)에서의 전파 신호 상에 구현될 수 있는 컴퓨터 프로그램 전파 신호 제품으로서 구현될 수 있다. 그러한 반송파 매체 또는 신호는 도 2a의 본 사용자 디바이스 승인 시스템(100)에 소프트웨어 명령어들 중 적어도 일부를 제공한다.

[0045] 본 발명의 특정 예시적 실시예들은 디바이스가 어떻다고 말하는 그대로이고 요청되는 바에 따라 정확하게 명령어들을 실행시키도록 신뢰될 수 있을 때, 온라인 서비스들이 상당히 강화될 수 있다는 전제에 기반한다. 서비스 제공자는 서비스 제공자의 서버들이 관리상의 제어 하에 있고 및 통상적으로 물리적으로 보호되므로, 일반적으로 서비스 제공자의 서버들에 확신을 갖는다. 그러나, 서비스 제공자의 서비스들의 거의 모두는 서비스 제공자가 매우 조금 인지하고 서비스 제공자가 거의 어떤 제어도 가하지 않는 디바이스들을 통해 사용자들에게 전달된다.

[0046] 신뢰된 실행 기술의 사용을 통해, 특정한 본 발명의 실시예들은 소비자 디바이스들의 알려지지 않은 세계에서의 신뢰의 위안처를 서비스 제공자에게 제공할 수 있다. "이것에 서명하시오", 또는 "이것을 복호화하시오"와 같은 기본 능력들은 주 OS의 어두운 세계 외부에서 실행된다. 키들은 메모리에 결코 노출되지 않고 생성되고 적용될 수 있고 디바이스 제조자로 다시 추적되는 보증들의 체인을 통해 입증될 수 있다.

[0047] 본 발명의 특정 양태들은 디바이스들의 신뢰를 가능하게 한다. 일부 실시예는 디바이스와의 신뢰 가능한 관계가 최종 사용자와의 훨씬 더 안전하고, 더 용이하고, 더 강한 관계에 기여할 수 있다는 기본적 전제 상에서 운용된다. 이를 성취하는 것은 현재의 트랜잭션에 수반되는 디바이스가 앞선 트랜잭션들에서의 것과 동일한 디바이스라는 확신을 인지하는 것을 필요로 한다. 이는 또한 복호화 또는 서명과 같은 민감한 작동들을 수행하는 것이 요청되면, 디바이스가 보호된 정보를 누설하지 않을 것이라는 보장을 필요로 한다.

[0048] 하나의 예시적 바람직한 실시예는 신뢰된 실행 환경(TEE)에서 실행되는 디바이스 코드를 포함한다. TEE는 바람직하게는 주 OS 외부의 작은 애플릿들을 실행시키는 하드웨어 환경이다. 이는 디바이스 제조자로 시작하여 보증들의 에코시스템에 의해 통제되는 특별히 만들어진 하드웨어로 멀웨어 또는 스누핑으로부터 민감한 코드 및 데이터를 보호한다.

[0049] 디바이스 무결성 입증/승인 - 일부 예시적 실시예

[0050] 도 2a는 구성 요소들(200)을 갖는 본 발명에 따른 예시적 디바이스 승인 시스템을 도시하는 블록도이다. 이러한 시스템 구성 요소들(200)로, 웹 개발자들 및 앱 개발자들은 애플리케이션 프로그램 인터페이스(API)를 통해 엔드포인트 사용자 디바이스들(205)에서의 견고해진 암호화 및 신원 키들을 이용할 수 있다. 게다가, 추가 서비스들이 디바이스 관리, 백업, 입증 등을 위한 이러한 시스템 구성 요소들(200)에 의거하여 제공될 수 있다. 이러한 시스템을 지원하기 위해, 신원 키들의 등록, 및 입증, 백업 및 디바이스 그룹화를 위한 디바이스 관리 서비스들의 세트가 관리된다.

[0051] 바람직한 예시적 실시예에서, 통상적 접근법들에서와 같이 미션 크리티컬 데이터를 유지 관리하지 않고, 오히려 서비스 제공자들(204)과 사용자 디바이스들(205) 사이의 한결같은 한층 더 매우 보안적 연결들에 대한 플랫폼을 제공하는 것이 시스템의 의도일 것이다. 시스템의 한편은 사용자 디바이스(205)에 대한 명령어를 마련하는 인코더(210)이고 다른 한편은 그러한 명령어에 따라 작동할 수 있는 신뢰된 실행 환경(TEE) 애플릿(208)인 디바이스 Rivet이다. 본 발명의 일 실시예에 따른 프로토콜은 이러한 명령어들 및 답신들이 구성되는 방법을 한정한다.

[0052] 디바이스 Rivet 또는 TEE 애플릿(208)은 바람직하게는 물리적인 작업과 디지털 작업 사이의 혁신적인 결속을 구현한다. 디바이스 Rivet 또는 TEE 애플릿(208)은 디바이스(205)의 하드웨어에 신원, 트랜잭션 및 입증의 특징부들을 락킹한다.

[0053] 도 2b에 도시된 본 발명의 일 실시예에 따른 시스템(200)은 모든 디바이스와의 영속적 연결을 유지하기 위해 보안적 소켓을 사용할 수 있다. 이러한 채널은 페어링 및 다른 관리상의 기능들에 사용된다. 라이브러리 코드(209)는 명령어의 구성 및 서명을 단순화하기 위해 서비스 제공자들에게 제공될 수 있다. 이러한 라이브러리(209)는 예를 들어, Python 같은 동적 의미론을 갖는 객체 지향적 높은 레벨 프로그래밍 언어와 같은 프로그래밍 언어로 구현될 수 있다.

[0054] 하나의 예시적 바람직한 실시예에서, TEE는 Rich 운영 체제와 함께 실행되고 그러한 Rich 환경에 보안 서비스들을 제공하는 모바일 전화기 하드웨어 보안 칩 별도 실행 환경으로서 구현될 수 있다. TEE는 Rich OS보다 더 높은 레벨의 보안을 제공하는 실행 공간을 제공한다. 다른 예시적 실시예에서, TEE는 가상 기계로서 구현될 수 있다. 보안 요소(SE)(aka SIM)만큼 보안적이지는 않지만, TEE에 의해 제공되는 보안은 일부/많은 애플리케이션에 충분하다. 이러한 방식으로, TEE는 SE보다 상당히 더 낮은 비용으로 Rich OS 환경보다 더 높은 보안을 가능하게 하는 균형을 전할 수 있다.

[0055] 링 관리자(212)는 사용자 디바이스들(205)의 컬렉션들 (또는 링들)을 관리하기 위해 최종 사용자들에게 제공되는 서비스로서 구현될 수 있다. 디바이스들(205)은 단일 신원으로 그룹화되고 서로 백업하고 보증하는데 사용될 수 있다. 링들은 디바이스들의 네트워크를 생성하기 위해 다른 링들과 연관될 수 있다. 일부 바람직한 실시예들에서, 링들은 (새로운 키와는 대조적으로) 개인 디바이스 공개 키들의 컬렉션이다. 환경에 많은 공유된 디바이스가 없으면, 바람직하게는 디바이스들의 목록은 바람직하게는 증가된 계산 및 대역폭 리소스들에 대한 가능성이 디바이스 목록 상의 공개 키들 모두를 갖는 메시지를 암호화하기 위해 시간 소비를 쏟고 도입시킬 수 있으므로, 짧을 수 있다.

[0056] 바람직하지 않은 예시적 실시예에서, 링은 디바이스(205)의 고유 개인 키 외의 공유된 개인 키로서 구현될 수 있다. 그러나, "개인 키"를 공유하는 것이 전형적이지도 않고, 오래 지속된 공유된 대칭 키를 갖는 것이 바람직하지도 않을 것이라는 점이 주목되어야 한다.

[0057] 본 발명의 일 실시예에 따른 시스템의 일 양태는 디바이스를 등록하고 디바이스에 서비스 제공자의 키들을 구비시킨다. 본 발명의 API들은: 신뢰 가능하고 익명의 디바이스 id를 얻는 것을 포함하는 다수의 민감한 디바이스 측 트랜잭션의 보안적 실행을 가능하게 하며 - 요청 시에, 본 발명의 일 실시예는 디바이스에 대한 서명 키를 생성할 것이다. 공개 키는 디바이스를 식별하고 이것과 통신하는데 사용될 수 있는 스트링으로 해싱된다. 개인 키는 하드웨어에 락킹되게 유지되고 ID를 요청했던 서비스만을 대신하여 적용될 수 있으며; 디바이스가 어떤 것에 서명하게 하는 - 디바이스 신원의 개인 키는 이러한 특정 디바이스가 수반되었다는 것을 입증하는 것들에 서명하는데 사용될 수 있다. 서명 세레모니는 키가 디바이스의 정상적 처리 환경에 결코 노출되지 않도록 보안적 하드웨어에서 실행되며; 디바이스가 어떤 것을 암호화하게 하는 - 암호화 키는 요청 시에 생성되고 임의의 블랍의 데이터에 적용될 수 있다. 암호화 및 복호화는 국부적으로 트리거되고 키를 보호하기 위해 보안적 실행 환경 내에서 일어나며; 비트코인 계정을 생성하는 - 디바이스는 TEE로 구축되는 난수 발생기(RNG)를 사용하여 새로운 비트코인 계정을 생성하도록 요청될 수 있으며; 비트코인 트랜잭션에 서명하는 - 디바이스는 트랜잭션에 서명하기 위해 디바이스의 사적 비트코인 계정 키를 적용시키고 그 다음 트랜잭션을 서비스 제공자에게 복귀시킬 수 있으며; 확인을 보안하는 - 더 새로운 TEE 환경들은 신뢰된 실행에 더하여 신뢰된 디스플레이 및 입력을 지원한다. 신뢰된 디스플레이는 "트랜잭션 총계를 확인하시오"와 같은 단순한 확인 메시지가 최종 사용자에게 제공되는 것을 가능하게 하며; 신원들을 공유하고 백업하도록 디바이스들을 가입시키는 - 대부분의 사용자는 수개의 디바이스를 갖는다. 본 발명의 특정 실시예들은 다수의 디바이스가 링으로 결속되는 것을 가능하게 하므로, 다수의 디바이스는 사용자를 대신하여 서비스 제공자에게 다수의 디바이스 자체를 상호 교환 가능하게 제공할 수 있다.

[0058] 서비스 제공자는 디바이스에서 하드웨어 키들을 생성하기 위해 제3 자 에이전트/프로세스를 호출한다. 상이한 타입들의 키들은 암호화 코인들 또는 데이터 암호화를 위해서와 같은 목적에 따라 이용 가능하다. 하드웨어 키들은 생성 동안 확립되는 단순한 사용 규칙들에 의해 통제된다. 예를 들어, 키는 사용 요청들이 키를 생성했던 서비스 제공자에 의해 서명되거나, 사용자가 신뢰된 사용자 인터페이스(TUI)를 통해 액세스를 확인하는 것을 필요로 할 수 있다.

[0059] 디바이스 Rivet(208)은 디바이스(205)와 "페어링되었던" 서비스 제공자(204)로부터의 명령어에만 응답할 것이다. 승인 웹 사이트(206)는 디바이스 및 서비스 제공자 둘 다의 무결성 및 신원을 확인할 수 있으므로, 페어링 세레모니를 행한다. 디바이스(205)가 페어링될 때, 디바이스(205)는 서비스 제공자(204)의 공개 키를 획득하는 반면에, 서비스 제공자는 디바이스(205)에 대해 고유하게 생성된 신원 및 공개 키를 얻는다.

[0060] 제3 자 에이전트/프로세스가 국부 호출들을 지원하지만, 이상적으로 모든 명령어는 서비스 제공자(204)에 의해 서명된다. 이는 악성 애플리케이션에 의해 적용되는 것으로부터 디바이스 키를 보호한다. 인코더(210)는 애플리케이션 서버 상의 디바이스 명령어들을 마련하고 이것들에 서명하는 것을 돕도록 제공된다.

[0061] 앱들의 근원의 강한 보장 및 다른 앱들의 실행으로부터의 불투명한 구분으로부터 크게 이익을 얻을 수 있는 부류의 앱들이 있다. 이는 신뢰된 실행 환경 또는 TEE로서 알려져 있다. 주 OS 및 메모리 스택 상에서 실행되는 앱과는 달리, TEE에서 실행되는 앱은 OS에 의한 스누핑 없이 발휘될 수 있는 암호화 프리미티브들에 대한 액세스를 갖는다. 일정 플랫폼들에서, 앱은 디바이스의 작동자와의 사적 상호 작용을 보장하기 위해 사용자 입력 및 디스플레이에 대한 직접적 액세스를 또한 갖는다. 기술이 10 년이 훨씬 넘는 동안 계속되었지만, TEE에 대한 지원을 갖는 디바이스들이 이용 가능해졌던 것은 단지 최근이다. 예를 들어, Intel은 2011 년에 상업 솔루션들의 납품을 시작하였고 ARM 합작 기업인, Trustonic은 2013 년에 런칭되었다.

[0062] TEE로 애플릿을 전개시키는 것은 전용 하드웨어 디바이스를 납품하는 것과 유사하다. 실행 및 데이터는 호스트의 임의의 다른 기능으로부터 암호학적으로 격리된다. 신뢰된 실행 기술의 대부분의 애플리케이션이 기업 보안 또는 DRM과 관련되었지만, 본 발명의 일 실시예는 대신에 통상의 웹 서비스들의 요구들에 집중되는 애플릿을 제공한다. 비트코인과 같은 암호화 화폐들은 소비자 키 보안에 대한 요구를 강조하였다.

[0063] 본 발명의 일 실시예는 호출들을 보안적 환경으로 옮기는 네이티브 API를 제공한다. 상이한 TEE 환경들이 매우 상이한 아키텍처들을 따르지만, 본 발명의 일 실시예의 API는 애플리케이션에 균일한 인터페이스를 제공하도록 설계된다.

[0064] 모든 TEE 애플릿으로와 같이, 본 발명의 실시예들에 따른 TEE 애플릿들은 신뢰된 애플리케이션 관리자 또는 TAM 없이 설치되고 초기화될 수 없다. TAM은 인증 기관(CA)과 유사한 역할을 한다. TAM은 디바이스 제조자와의 관계를 보안하고 또한 디바이스로 로딩될 수 있는 모든 애플릿에 서명한다. 이러한 방식으로, TAM은 애플릿 및 TEE 둘 다의 출처 및 무결성에 대한 보장을 표현한다.

[0065] 디바이스 무결성 입증

[0066] 본 발명의 실시예들은 블록 체인 트랜잭션 상에 서명한 것으로서의 인지된 상태에 대한 디바이스 무결성의 보장을 자동화함으로써 디바이스 무결성 입증을 제공한다. 본 발명의 일 실시예에 의해 구현되는 시스템은 도 2c에 도시된 수개의 구성 요소로 구성된다. 디바이스 어댑터(220)는 인터페이스를 서비스 제공자(204) 애플리케이션에 제공하고 디바이스 TEE(208)와 통합되는 엔드포인트 디바이스 상에서 실행되는 소프트웨어 서비스이다. 신뢰된 실행 환경(TEE - 때때로 TrEE)은 Rich OS와 함께 실행되고 그러한 Rich 환경에 보안 서비스들을 제공하는 모바일 전화기 하드웨어 보안 칩 별도 실행 환경이다. TEE는 Rich OS보다 더 높은 레벨의 보안을 제공하는 실행 공간을 제공하며; 보안 요소(SE)(aka SIM)만큼 보안적이지는 않지만, TEE에 의해 제공되는 보안은 일부/많은 애플리케이션에 충분하다. 이러한 방식으로, TEE는 SE보다 상당히 더 낮은 비용으로 Rich OS 환경보다 더 높은 보안을 가능하게 하는 균형을 전한다. 다른 구성 요소인, 디바이스 TEE(208)는 하드웨어 보안된 TEE에서 실행되는 소프트웨어 프로그램이다. 디바이스 TEE(208)는 멀웨어 또는 심지어 디바이스 작동자로부터의 위해 없이 암호화 기능들을 실행시키도록 특별히 설계된다. 다른 구성 요소인, 디바이스 등록기(221)는 디바이스를 블록 체인(222)으로 등록하는 서비스이다. 블록 체인(222)은 디바이스 등록 및 속성들을 저장하고 트랜잭션들을 실행시키는 것 둘 다를 위해 사용된다. 상이한 블록 체인들이 있을 수 있다. 다른 지원하는 구성 요소는 디바이스와의 트랜잭션을 행할 것을 요구하는 애플리케이션인 서비스 제공자(204)이다. OEM(주문자 생산 방식)(223)은 디바이스를 구축하는 엔티티 및/또는 디바이스의 출처에 대해 암호로 보증하도록 인증되는 신뢰된 애플리케이션 관리자(TAM)이다.

[0067] 본 발명의 일 실시예에 따르면, 도 2c 소프트웨어로 도시된 디바이스 어댑터(221)가 처음으로 실행될 때, 디바이스 어댑터(221)는 공개/개인 키 쌍을 생성할 것을 디바이스 TEE(208)에 요청할 것이다. 공개 키는 디바이스 제조 동안 확립되는 보증 키에 의해 서명된다. 이러한 서명된 공개 키는 디바이스 등록기(221)로 송신되고 OEM(223)으로 입증된다. 등록은 디바이스 작동자로부터의 확인을 수반할 수 있다. 등록은 판매원의 존재에서 판매하려는 순간에 보증을 수반할 수 있다. 등록기는 이하의 것: 부트 프로세스에 의해 생성되는 플랫폼 구성 레지스터들(PCR's)의 합성값, 바이오스 버전, OS 버전, GPS 위치 중 하나 이상을 포함하는 디바이스 측정 기록을 디바이스에 요청할 수 있다. 이러한 데이터는 디바이스 개인 키에 의해 서명된다. 이러한 데이터는 등록기에 의해 추가로 서명된다. 결과로서 생기는 데이터 세트는 장래 무결성 체크들에 대한 금전 기준 또는 참조값이 된다. 디바이스 작동자로부터의 확인은 금전 기준 또는 참조값을 수집하는데 필요할 수 있다. 이러한 데이터 세트는 공개 암호화 원장으로 발송된다. 공개 기록은 등록기의 보증에 따른 등록의 시간의 암호화 입증에 의해 확립된다. 등록은 위치 또는 기업 이름 또는 디바이스 형식/모델과 같은 속성 데이터를 더 포함할 수 있다. 등록은 등록 시의 등록기의 정책 규정들을 준비한 서명된 문서를 참조할 수 있다. 디바이스 등록기(221) 또는 다른 신뢰된 무결성 서버는 블록 체인 상의 다중 시그니처 트랜잭션의 서명한 것으로서 참조될 수 있는 블록 체인 계정 키(공개/개인 키 쌍)를 생성한다. 블록 체인 트랜잭션에 나타내어지는 서명한 값은 등록기에 의해 공동 서명되지 않는다면, 소비되거나 전송될 수 없다.

[0068] 트랜잭션에 서명하기 위해, 무결성 서버는 디바이스로부터의 최근의 측정치를 기대한다. 이러한 측정치는 디바이스 어댑터에 직접 요청되거나 디바이스와의 영속적 소켓 연결을 통해 서버에 의해 불러와질 수 있다. 현재의 측정치는 블록 체인에서의 금전 측정치 또는 참조값에 대하여 비교된다. 측정치들이 부합하면, 트랜잭션이 서명된다. 측정치들이 부합하지만, 최근의 측정치가 지정된 시간 윈도우보다 더 오래되었으면, 요청이 거절된다. 측정치들이 부합하지 않으면, 요청이 거절된다. 거절이 있으면, 트랜잭션은 거절을 무시하도록 요청될 수 있는 다른 수작업의 서명한 것으로 마련되었을 수 있다. 측정치들이 부합하지 않으면, 디바이스는 새로운 측정치가 수집되는 등록 재개를 겪게 될 수 있다. 측정치가 부합할 때마다, 디바이스 등록 기록은 성공 카운트로 업데이트될 수 있다. 무결성 서버는 문제가 다른 부합하는 측정치들 또는 속성들을 고려하여 심각한 것으로 간주되지 않으면, 부합하지 않는 측정치를 수락할 정책 규칙들이 주어질 수 있다.

[0069] 본 발명의 일 실시예에 따른 시스템은 측정치들을 부합시키고 트랜잭션에 서명하는 작업을 행하기 위해 무결성 서버보다는 오히려 신뢰된 디바이스들의 컬렉션으로 구현될 수 있다. 시스템은 Ethereum에 의해 개발되는 시스템과 같은 스마트 블록 체인 시스템으로 구축되는 특징부들을 사용하여 트랜잭션 처리 동안 직접 무결성 측정치

들을 부합시킬 수 있다.

[0070] 디바이스 무결성 입증 - 승인 웹 사이트

[0071] 예시적 실시예에서, 승인 웹 사이트(206)는 디바이스들(205) 및 서비스 제공자들(204)의 신원 키들을 등록시키기 위해 제3 자 에이전트/프로세스 개인 키를 사용하는 Python으로 기록되는 JSON API일 수 있다. 등록 동안, 사용자 디바이스(205) 또는 서비스 제공자(204)의 공개 키는 TEE 애플릿(208)에 의해 기록된다. 등록은 TEE 애플릿(208)이 디바이스(205)를 서비스 제공자(204)와 페어링하는 것을 가능하게 한다. 페어링의 결과는 사용자 디바이스(205)가 제3 자 에이전트/프로세스에 의해 보증되는 서비스 공개 키를 갖고 그러므로 서비스 제공자(204) 명령어들에 응답할 수 있는 것이다.

[0072] 본 발명의 일 실시예에 따른 프로토콜은 명령어를 수락하도록 디바이스(205)에 적용되어야 하는 서명/암호화 및 명령어의 구조체를 지정한다. 명령어 그 자체는 예를 들어, 명령어 코드, 버전 데이터 및 페이로드를 포함하는 C 구조체로서 마련될 수 있다. 전체 구조체는 바람직하게는 서비스 제공자 키에 의해 서명되고 디바이스 국부 커맨드를 호출함으로써 디바이스 TEE 애플릿(208)에 전달된다.

[0073] 바람직하게는, 모든 사용자 디바이스(205)는 고유 신원 크리덴셜들을 제공해야 한다. 디바이스들은 단독의 엔티티로서의 역할을 하도록 링에 가입할 수 있다. 일 실시예에서, 디바이스(205)는 목록으로서 국부적으로 저장되지만, 공개적으로 크로스 플랫폼 승인으로 변환되는 그룹 ID들을 지원할 수 있다. TEE 어댑터(216)는 TEE로 개재되는 디바이스 Rivet/TEE 애플릿(208)과 파트너 앱들 및 온라인 서비스들의 외부 세계 사이의 인터페이스로서 구성될 수 있다. 구현에서, TEE 어댑터(216)는 디바이스들, 하드웨어 지원 및 OS 아키텍처에 걸친 기본 능력들에 의해 적어도 부분적으로 결정될 하나 이상의 다양한 형태로 나타날 수 있다.

[0074] 디바이스 무결성 입증 - 승인 시스템 어댑터

[0075] 승인 시스템 어댑터(214)는 도 2d에 도시된 바와 같이 밖으로 그리고 안으로 본 인터페이스들로 구성된다. 안으로 본 인터페이스인, TEE 어댑터(216)는 디바이스 Rivet(208)과의 전매 통신을 처리한다. 호스트 어댑터(217)는 제3 자 애플리케이션들에 서비스들을 노출시키도록 제공된다. 호스트 어댑터(217)는 브라우저들 또는 시스템 서비스들과 같은 상이한 국부 콘텍스트들을 통해 승인 시스템 어댑터(214)의 인터페이스를 제공한다. 초기에 이는 안드로이드 서비스 및 Windows 기업 프로세스일 수 있지만, 다양한 콘텍스트에 대한 다수의 실현이 예상된다. 소켓 어댑터(215)는 클라이언트 환경 승인 웹 사이트(206)에 연결된다. TEE 어댑터(216) 구성 요소는 커맨드들을 디바이스 Rivet(208)으로 송신하는 전매 글루(glue)이다. 안드로이드 구현에서, 승인 시스템 어댑터(214)는 안드로이드 NDK 서비스 앱으로서 나타날 수 있고 부팅에서 런칭하도록 구성될 수 있다. 승인 시스템 어댑터(214)는 디바이스 Rivet(208)으로 송신되는 메시지 버퍼들을 마련하고 그 다음 응답 이벤트의 통지를 동기적으로 대기한다. 호스트 어댑터(217)는 주로 호스트 환경에서 TEE 어댑터(216)를 격리시키는 곳에 있다. 호스트 어댑터(217)는 잠재적으로 반대하는 환경에서 작동한다. 그러므로, 전형적으로 클라이언트가 위해되지 않았다는 제한된 보장이 있을 것이다. 그러므로, 호스트 어댑터의 역할은 주로 디바이스 Rivet(208)에 대한 쉬운 액세스를 용이하게 하는 것이다. 디바이스 Rivet(208)에 대해 의도되는 서비스 제공자(204)로부터의 명령어들은 서비스 제공자(204)에 의해 서명되고 그 다음 TEE 어댑터(216) 및 디바이스 Rivet(208)으로 패스될 것이다.

[0076] 디바이스에 등록되는 제1 서비스 제공자

[0077] 예시적 실시예에 따르면, 승인 웹 사이트(206)는 디바이스(205)에 등록되는 제1 서비스 제공자이다. 승인 웹 사이트(206)는 부가 서비스 제공자들을 그러한 디바이스(205)와 페어링할 수 있는 특별한 능력을 갖는다. 승인 웹 사이트(206)와의 통신은 웹 API를 통해 처리될 수 있고 승인될 것이다. 일 예에서, 이는 API 키로 구현된다. 바람직한 예시적 실시예에서, 이는 SSL 키 스왑을 사용하여 구현된다. 일부 실시예들에서, 모든 요청이 서명될 것이다.

[0078] 디바이스들과의 관계는 개인 키로 명령어들에 서명할 수 있는 것에 의존할 수 있다. 그러한 개인 키는 매우 민감하고 보호된다. 바람직하게는, 개인 키는 HSM에 넣어진다.

[0079] 일부 실시예들에서, 다수의 키가 사용되어, 하나의 키가 위해되면, 전체 시스템이 손실되지 않는다. 이는 예를 들어, 어느 디바이스들이 위해된 키와 연결되는지를 공격자가 아는 것을 더 어렵게 할 것이다. 더욱이, 시스템(200)은 바람직하게는 키들의 빈번한 회전을 용이하게 할 수 있는 도 2c에 도시된 소켓 어댑터(215)를 통해 모든 디바이스(205)와 거의 일정하게 연락한다.

[0080] 승인 웹 사이트(206)는 수개의 하위 구성 요소를 포함할 수 있다. 디바이스 ID는 승인 웹 사이트(206) 또는 다

른 등록 에이전트에 의해 디바이스에 할당되는 UUID로의 고유 식별자이다. 단명하는 포인터인, 디바이스 포인터는 임의의 국부 애플리케이션에 의해 요청될 수 있는 디바이스(150)에 제공될 수 있다. 디바이스 포인터는 승인 웹 사이트(206)에 대한 현재의 소켓 세션을 식별할 수 있고 그러므로 디바이스 통신 채널을 확립하고 영구적 식별자인, 디바이스 ID를 검색하는데 사용될 수 있다. 디바이스 등록의 루트는 고유한 익명의 식별자, 등록 날짜, 디바이스 하드웨어에 유지되는 개인 키와 페어링된 공개 키 및 등록 에이전트로부터의 보증 시그니처를 포함한다. 이러한 정보는 디바이스 등록 기록에 기록된다. TEE 애플릿(208)은 물리적인 작업과 디지털 작업 사이의 결속을 구현한다. 디바이스 Rivet(209)은 하드웨어에 신원, 트랜잭션 및 입증의 특징부들을 락킹한다.

[0081] 명령어들을 처리하는 프로토콜

[0082] 디바이스 Rivet(209)의 대응물은 인코더(210)이다. 인코더(210)는 서비스 제공자(204)에 의해 서명되고/되거나 암호화되는 특정 디바이스에 의해 실행될 커맨드를 마련한다. 서비스 제공자 공개 키들은 승인 웹 사이트(206)에 의해 행해지는 페어링 프로세스 동안 디바이스로 미리 로딩된다. 이는 디바이스 Rivet(209)이 요청의 근원을 입증하고, 필요하다면, 명령어의 내용들을 복호화하는 것을 가능하게 한다. 명령어를 패키징하고 전달하는 시퀀스가 도 3a에 도시된다. 서비스 제공자(204)는 인코더(210) 라이브러리들의 도움으로 명령어 기록을 생성한다. 명령어는 타입, 타겟 디바이스 및 페이로드를 포함한다. 명령어는 디바이스 키로 인코딩될 수 있고 서비스 제공자 키에 의해 서명되어야 한다. 디바이스 키는 디바이스 등록 기록을 검색함으로써 승인 웹 사이트(206)로부터, 또는 블록 체인으로부터 직접 불러와진다.

[0083] 디바이스를 등록하는 프로토콜

[0084] 블록 체인 상의 디바이스에 대한 버스 증명서의 디바이스 등록 또는 생성은 본 발명의 예시적 실시예들에 필수적이다. 도 3b에 도시된 등록 프로세스는 사용자에게 성가시지 않거나 투명하기도 해야 한다. 이상적으로, 충분히 훌륭한 디바이스 ID는 핀 또는 다른 메모리 테스트로의 디바이스와 사용자 사이의 관계의 개인화뿐만 아니라; 예를 들어, 판매원의 존재에서 디바이스를 등록함으로써 사용자와 디바이스 사이의 합법적 결속을 포함할 것이다. 충분히 훌륭한 디바이스 ID는 출처를 보장하기 위해 디바이스를 제조했던 OEM의 보증 키들을 검색할 것이다. 충분히 훌륭한 디바이스 ID는 디바이스 등록의 목적, 권한 및 익명성에 트레이닝하는 것을 포함할 수도 있다. 사람은 투명하게 ID를 생성하는 것으로 막 시작할 수 있다. 등록의 맥락에서 이러한 변동성 때문에, 등록 에이전트는 신뢰가 끝나는 곳에서 신뢰가 연장되고 있다는 것을 보장하기 위해 등록의 콘텍스트를 기록해야 한다. 예를 들어, OEM 보증 키를 테스트하는 것은 디바이스 Rivet이 적절한 TEE에서 작동하고 있다는 것을 대단히 더 확실하게 한다.

[0085] 도 2c에 도시된 예시적 실시예에서, 디바이스 어댑터(220) 소프트웨어가 처음으로 실행될 때, 디바이스 어댑터(220) 소프트웨어는 공개/개인 키 쌍을 생성할 것을 디바이스 TEE(208)에 요청할 것이다. 공개 키는 디바이스 제조 동안 확립되는 보증 키에 의해 서명된다. 이러한 서명된 공개 키는 디바이스 등록기(221)로 송신되고 OEM(223)으로 입증된다. 등록은 디바이스 작동자로부터의 확인을 수반할 수 있거나 등록은 판매원의 존재에서 판매하려는 순간에 보증을 수반할 수 있다. 등록기(221)는 이하의 것: 부트 프로세스에 의해 생성되는 플랫폼 구성 레지스터들(PCR's)의 합성값, 바이오스 버전, OS 버전, GPS 위치, 바이오스 식별자, 네트워크 인터페이스 식별자, 파일의 수, 파일들의 크기, 딕셔너리들, 색인들 및 데이터/탐색 트리 구조체들과 같은 디바이스에 대한 속성들, 디바이스의 수를 식별하는 프로세서, 또는 다른 그러한 정보 중 하나 이상을 포함하는 디바이스 측정 기록을 디바이스에 요청할 것이다. 이러한 데이터는 디바이스 개인 키에 의해 서명되고 등록기(221)에 의해 추가로 서명될 수 있다. 결과로서 생기는 데이터 세트는 장래 무결성 체크들에 대한 금전 기준이 된다. 디바이스 작동자로부터의 확인은 금전 기준을 수집하는데 필요할 수 있다. 이러한 데이터 세트는 네임코인과 같은 공개 암호화 원장으로 발송된다. 공개 기록은 등록기의 보증에 따른 등록의 시간의 암호화 입증에 의해 확립된다. 등록은 위치 또는 기업 이름 또는 디바이스 형식/모델과 같은 다른 속성 데이터를 더 포함할 수 있다. 등록은 등록 시의 등록기의 정책 규정들을 준비한 서명된 문서를 참조할 수 있다. 디바이스 등록기(221) 또는 다른 신뢰된 무결성 서버는 블록 체인 상의 다중 서명 트랜잭션의 서명한 것으로서 참조될 수 있는 블록 체인 계정 키(공개/개인 키 쌍)를 생성한다. 블록 체인 트랜잭션에 나타내어지는 서명한 값은 등록기(221)에 의해 공동 서명되지 않는다면, 소비될/전송될 수 없다. 트랜잭션에 서명하기 위해, 무결성 서버는 디바이스로부터의 최근의 측정치를 기대한다. 이러한 측정치는 디바이스 어댑터에 직접 요청되거나 디바이스와의 영속적 소켓 연결을 통해 서버에 의해 불러와질 수 있다. 현재의 측정치는 블록 체인에서의 금전 측정치에 대하여 비교된다. 측정치들이 부합하면, 트랜잭션이 서명되며, 측정치들이 부합하지만, 최근의 측정치가 지정된 시간 윈도우보다 더 오래되었으면, 요청이 거절된다. 측정치들이 부합하지 않으면, 요청이 거절된다. 거절이 있으면, 트랜잭션은 거절을 무시하도록 요청될 수 있는 다른 수작업의 서명한 것으로 마련되었을 수 있다. 측정치들이 부합하지 않으면, 디바이

스는 새로운 측정치가 수집되는 등록 재개를 겪게 될 수 있다. 측정치가 부합할 때마다, 디바이스 등록 기록은 성공 카운트로 업데이트될 수 있다. 무결성 서버는 문제가 다른 부합하는 측정치들 또는 속성들을 고려하여 심각한 것으로 간주되지 않으면, 부합하지 않는 측정치를 수락할 정책 규칙들이 주어질 수 있다. 이러한 시스템은 측정치들을 부합시키고 트랜잭션에 서명하는 작업을 행하기 위해 무결성 서버보다는 오히려 신뢰된 디바이스들의 컬렉션으로 구현될 수 있다. 이러한 시스템은 Ethereum에 의해 개발되는 시스템과 같은 스마트 블록 체인 시스템으로 구축되는 특징부들을 사용하여 트랜잭션 처리 동안 직접 무결성 측정치들을 부합시킬 수 있다.

[0086] 블록 체인 상의 디바이스에 대한 버스 증명서

[0087] 일 실시예는: 사용자 디바이스로 락킹되는 공개/개인 키 쌍을 생성함으로써 사용자 디바이스에 대한 디바이스 신원을 확립하는 단계; 디바이스의 제조 또는 생성, 디바이스의 실행 환경의 제조 또는 생성, 및/또는 디바이스 상의 애플리케이션의 제조 또는 생성 동안 확립되는 보증 키에 의해 디바이스의 공개 키의 서명하는 단계; 및 신뢰된 제3 자에 디바이스를 등록하는 단계로서: 디바이스로부터 생성된 공개 키를 요청하고 얻는 단계; 디바이스 플랫폼 구성 레지스터들(PCR), 바이오스, OS 및/또는 GPS와 관련되는 속성들을 포함하는 디바이스의 디바이스 측정 기록을 요청하고 얻는 단계; 제3 자 및 디바이스에 의한 디바이스 측정 기록의 보증하는 단계; 및 블록 체인으로 디바이스를 등록하는 단계로서, 공개 암호화 원장으로 보증된 디바이스 측정 기록을 발송하는 단계; 및 블록 체인 상의 다중 시그니처 트랜잭션에서 서명한 것으로서 참조될 수 있는 블록 체인 계정 키 쌍을 생성하는 단계를 포함하는 단계를 포함하는 단계를 포함하는 블록 체인 통신 네트워크에서의 사용자 디바이스에 대한 버스 증명서를 생성하는 방법일 수 있다. 일부 실시예들에서, 방법은 디바이스와 페어링할 것을 요구하는 제1 서비스 제공자의 요청에서 제3 자에 디바이스를 등록하는 단계를 포함할 수 있다. 일부 실시예들에서, 디바이스를 등록하는 단계는 서비스로서 제공될 수 있다. 디바이스에 의한 디바이스 측정 기록의 보증은 디바이스 개인 키에 의한 기록의 서명을 포함할 수 있다. 제3 자에 의한 디바이스 측정 기록의 보증은 서비스로서 제공될 수 있다. 등록은 등록 시의 등록 제공자의 정책 규정들을 준비한 문서의 서명을 더 포함할 수 있다. 공개 암호화 원장은 네임코인일 수 있다. 보증된 디바이스 측정 기록은 서비스 제공자와 디바이스 사이의 트랜잭션들에 대한 참조값을 확립할 수 있다. 게다가, 디바이스 작동자에 의한 확인은 디바이스로부터 디바이스 속성들의 디바이스 측정 기록을 얻는데 필요하다. 디바이스 속성들은 위치, 기업 이름 및/또는 디바이스 형식/모델을 더 포함할 수 있다. 게다가, 서비스 제공자와 디바이스 사이의 트랜잭션은 디바이스에 대한 확립된 참조값과 비교되는 디바이스 측정 기록을 생성하고 제공하는 디바이스를 필요로 할 수 있다. 다른 실시예들에서, 트랜잭션은 비교가 부합을 야기하면, 가능해지거나, 트랜잭션은 비교가 불부합을 야기하면, 거절되거나, 트랜잭션은 비교가 부합을 야기하고 디바이스에 의해 제공되는 기록이 지정된 시간 윈도우보다 더 오래되었으면, 거절되거나, 디바이스는 비교가 불부합을 야기하면, 디바이스의 버스 증명서를 재생성하는 것이 필요하다. 게다가, 블록 체인으로 디바이스를 등록하는 단계는 비교가 부합을 야기하면, 성공 카운트로 업데이트되는 디바이스 등록 기록을 생성하는 단계를 더 포함할 수 있다. 비교는 신뢰된 디바이스들의 컬렉션에 의해 구현될 수 있다. 비교를 수행하는 엔티티는 등록을 수행하는 엔티티와는 별도일 수 있다.

[0088] 다른 실시예는: 블록 체인 통신 네트워크; 블록 체인 네트워크에서의 사용자 디바이스; 신뢰된 제3 자; 및 사용자 디바이스에 대한 버스 증명서를 생성하는 시스템을 포함하는 시스템일 수 있으며, 상기 시스템은 사용자 디바이스로 락킹되는 공개/개인 키 쌍을 생성함으로써 사용자 디바이스에 대한 디바이스 신원을 확립하고; 디바이스의 제조 또는 생성, 디바이스의 실행 환경의 제조 또는 생성, 및/또는 디바이스 상의 애플리케이션의 제조 또는 생성 동안 확립되는 보증 키를 사용하여 디바이스의 공개 키에 서명하고: 디바이스로부터 생성된 공개 키를 요청하고 얻고; 디바이스 플랫폼 구성 레지스터들(PCR), 바이오스, OS 및/또는 GPS와 관련되는 속성들을 포함하는 디바이스의 디바이스 측정 기록을 요청하고 얻고; 제3 자 및 디바이스에 의한 디바이스 측정 기록을 보증하고; 공개 암호화 원장으로 보증된 디바이스 측정 기록을 발송하고, 블록 체인 상의 다중 시그니처 트랜잭션에서 서명한 것으로서 참조될 수 있는 블록 체인 계정 키 쌍을 생성함으로써, 블록 체인으로 디바이스를 등록함으로써; 신뢰된 제3 자에 디바이스를 등록하도록 구성된다.

[0089] 소유권 권한들을 누적시키기 위해 블록 체인 상의 트랜잭션들을 사용하는 것

[0090] 은행 계좌와 마찬가지로 비트코인 월렛 기능들은 비트코인들을 수신하고 저장할 뿐만 아니라 비트코인들을 비트코인 블록 체인에서의 전자 트랜잭션의 형태의 다른 비트코인들로 전송하는데 사용될 수 있다. 비트코인 어드레스는 사용자가 비트코인들을 받는 것을 가능하게 하는 고유 식별자이다. 비트코인들은 비트코인들을 비트코인 어드레스로 송신함으로써 전송된다. 비트코인 블록 체인에서의 트랜잭션들은 통상적으로 무료이다. 그러나, 다수의 어드레스를 사용하여 비트코인들을 송신하고 수신하는 트랜잭션들은 통상적으로 트랜잭션 수수료를 발생시

킬 것이다. 월렛은 사용자가 비트코인 어드레스들에 액세스할 수 있도록 개인 키들을 저장한다.

[0091] 블록 체인 상의 트랜잭션이 소유권 권한을 누적시키고 성취하는 시스템들 및 방법들이 제공될 수 있다.

[0092] 비트코인 트랜잭션이 새로운 라이센스 권한에 누적되는 서비스가 제공될 수 있다. 이는 권한에 누적되는 트랜잭션들의 체인을 식별할 트랜잭션 기록에서의 속성 정보와 스마트 약정을 통합시킴으로써 행해질 것이다. 궁극적으로, 이러한 권한은 본래 월렛 어드레스에 결속될 것이다. 특정 아이템이 구매될 때마다, 이러한 권한은 트랜잭션들의 누적이 블록 체인 상의 정보를 판독함으로써 빠르게 그리고 효율적으로 검증될 수 있다는 것을 보장하는 현재의 트랜잭션의 속성 데이터의 일부로서 지난 트랜잭션을 포함시킬 것이다. 블록 체인 상의 많은 작은 트랜잭션을 수행하는 행위는 계정이 소유권 권한 또는 재연 권한에 용이하게 누적되는 것을 가능하게 할 것이다. 특정 레벨이 도달되면, 누적이 중단될 것이고 영속적 권한이 블록 체인에 기록될 것이다.

[0093] 일부 실시예는 전자 트랜잭션들에 관여하기 이전에 디바이스 헬스를 입증하는 시스템들 및 방법들을 포함할 수 있다.

[0094] 이는 권한에 누적되는 트랜잭션들의 체인을 식별할 트랜잭션 기록에서의 속성 정보와 스마트 약정을 통합시킴으로써 행해질 것이다. 궁극적으로, 이러한 권한은 본래 월렛 어드레스에 결속될 것이다. 특정 아이템이 구매될 때마다, 이러한 권한은 트랜잭션들의 누적이 블록 체인 상의 정보를 판독함으로써 빠르게 그리고 효율적으로 검증될 수 있다는 것을 보장하는 현재의 트랜잭션의 속성 데이터의 일부로서 지난 트랜잭션을 포함시킬 것이다. 블록 체인 상의 많은 작은 트랜잭션을 수행하는 행위는 계정이 소유권 권한 또는 재연 권한에 용이하게 누적되는 것을 가능하게 할 것이다. 특정 레벨이 도달되면, 누적이 중단될 것이고 영속적 권한이 블록 체인에 기록될 것이다.

[0095] 비트코인 계정과 연관된 블록 체인 통신 네트워크에서의 트랜잭션들에 첨부되는 값을 누적시키는 시스템이 제공될 수 있으며, 시스템은 블록 체인 통신 네트워크; 블록 체인 네트워크에서의 전자 트랜잭션; 비트코인 계정; 비트코인 계정과 연관된 트랜잭션 기록; 블록 체인 네트워크에서의 전자 트랜잭션을 실행시키는 것의 일부로서 구현되는 트랜잭션 질의 프로세스를 포함한다. 구현은 계정과 연관된 앞선 트랜잭션의 존재에 대한 트랜잭션 기록의 체킹; 그리고 앞선 트랜잭션의 존재에 기반하여: 앞선 트랜잭션에 첨부되는 누적된 값을 얻고; 얻어지는 누적된 값을 증분하고; 증분된 누적된 값을 트랜잭션 기록에서의 트랜잭션에 첨부하고; 증분된 누적된 값을 트랜잭션에 인가하는 것을 더 포함할 수 있다.

[0096] 트랜잭션 질의 프로세스의 구현은 전자 트랜잭션을 실행시키는 발생되는 복수의 요금을 제로로 설정하는 것 및 미리 결정된 최대 누적된 트랜잭션값에 도달하거나 이것을 초과하는 증분된 누적된 값에 기반하여 계정과 연관된 권한의 성취를 나타내는 것을 더 포함할 수 있다.

[0097] 트랜잭션 질의 프로세스의 구현은 계정과 연관된 새로운 트랜잭션 기록을 생성하는 것; 및 새롭게 생성된 트랜잭션 기록에서의 성취된 권한의 표시를 저장하는 것을 더 포함할 수 있다.

[0098] 전자 트랜잭션은 특정 아이템과 연관될 수 있고, 계정과 연관된 트랜잭션 기록에서의 트랜잭션들은 암호화 보장을 갖는 체인을 형성하고, 트랜잭션 질의 프로세스의 구현은: 계정과 연관된 트랜잭션 기록에서의 기록된 지난 트랜잭션을 사용자가 질의하는 것을 가능하게 하는 것; 및 형성된 체인의 암호화 보장에 기반하여 특정 아이템에 대한 경비의 레벨을 계산하는 것을 더 포함할 수 있다.

[0099] 누적된 값을 트랜잭션에 인가하는 것은 성취된 권한을 암호화 키와 연관시키는 것; 키를 변경 방지 저장소에 저장하는 것; 성취된 권한과 연관된 누적된 값에 기여하는 트랜잭션들의 세트를 얻는 것; 및 누적된 값을 트랜잭션에 인가하기 이전에 트랜잭션들의 세트를 검증하는 것을 포함할 수 있다.

[0100] 일부 시스템에서, 트랜잭션들의 세트는 권한의 성취에 기여하기 위해 특정 기간 내에 완료되어야 한다. 성취된 권한은 특정 기간 후에 만료되고/되거나 권한의 사용의 결여에 기반하여 만료된다. 성취된 권한은 성취된 권한의 표시를 필요로 하는 부가 트랜잭션들의 구매를 가능하게 하기 위해 다중 시그니처 트랜잭션의 일부로서 사용된다.

[0101] 일부 시스템에서, 트랜잭션은 단일 아이템과 연관되고 2개의 성취된 권한을 수반하고 권한들과 연관된 누적된 값들은 단일 누적된 값을 야기하도록 암호로 병합된다.

[0102] 클라우드 서비스들 및 피어 서비스들에의 보장된 컴퓨터 명령어들

[0103] 컴퓨팅의 현재의 상태는 디바이스들이 Twitter 같은 클라우드 서비스에 연결되고 그 다음 뒤따라가는 데이터가

정확하다고 가정하는 승인 모델에 기반한다. 암호화된 전송이 통상적으로 사용되고 보장 모델이 데이터를 송신하는 전체 컴퓨터를 보장하는 것에 기반한다. 안티바이러스 및 무결성 입증 같은 기술들이 호스트 시스템에 대해 제공된다. 복합 시스템이 괜찮고 전달되는 중요 데이터를 신뢰할 것이라는 가정이 행해진다.

[0104] 승인은 컴퓨터 명령어들이 정확하다는 것을 보장하고 그 다음 처리를 위해 이러한 명령어들을 원격 서비스들로 전달하도록 원격 소스들 둘 다로부터의 국부 디바이스 내에 형성되는 보장된 이러한 명령어들로 증대될 수 있다. 시스템은 사용자 입력, 디바이스 입력, 원격 시스템 입력으로부터의 데이터를 수집하고 그 다음 이것이 수행되도록 의도된 트랜잭션인 것을 확인시키기 위해 보안적 메커니즘을 사용자에게 제공할 수 있다. 클라우드 서비스는 이러한 보장된 명령어를 수신하고 트랜잭션의 요소들이 정확한 것을 검증한다. 검증 프로세스는 트랜잭션이 처리에 수락되기 이전에 검증되는 국부 또는 원격 정책들을 도입시킬 수도 있다. 결과로서 생기는 데이터가 그 다음 로깅될 수 있다.

[0105] 범용 컴퓨팅 디바이스에서 전형적으로, 승인은 중대한 서비스들에 연결되는데 사용된다. 강한 승인으로도, 클라우드로 송신되는 정보가 사용자가 의도하는 정보라는 어떤 보장도 없다. 멀웨어는 데이터를 바꾸고 민감한 데이터의 절도 또는 위해를 야기하는 많은 방식을 찾아낼 수 있다. 본 발명의 목적은 제공되는 정보가 의도되는 데이터라는 것을 보장하기 위해 국부 및 원격 데이터 둘 다의 다수의 소스를 수집하는 것이다. 일정 데이터는 프로세스가 완료되었다는 것을 보장하도록 국부적으로 마스킹될 수도 있지만 상세한 사적 정보는 마스킹되게 유지된다. 서비스들은 그 다음 트랜잭션들이 의도된다는 것을 검증하고 사용자에 의해 제어되는 다수의 부가 프로세스 단계를 내부적으로 그리고 외부적으로 포함시킬 수 있다. 이는 트랜잭션이 정확하다는 것을 보장하도록 로깅 및 부가 검증을 보장할 수 있다. 이는 금융 시스템들에 뿐만 아니라 출입문 자물쇠들로부터 의료 디바이스들까지의 것들의 인터넷을 제어하는 데에도 사용될 수 있다.

[0106] 일부 시스템에서, 보안적 서브시스템은 다른 컴퓨터 시스템에의 전달에 대한 보안적 명령어를 모으는데 사용된다. 보안적 서브시스템은 시간, 위치, 신원, 준수 또는 다른 중요 데이터와 같은 부가 정보를 국부적으로 또는 원격으로 수집하고 첨부하고 명령어가 서명되고 그 다음 송신되기 이전에 명령어를 보안적으로 확인하는 메커니즘을 사용자에게 제공한다.

[0107] 일부 시스템에서, 보호된 명령어가 수신될 때, 보호된 명령어는 처리되기 이전에 검증된다. 검증은 국부적으로 또는 원격으로 행해질 수 있고 로깅 시스템들, 다른 중대한 프로세스 단계들, 위치 또는 시간으로부터의 부가 사용자 검증, 확인 또는 시그니처를 포함할 수 있다.

[0108] 일부 시스템에서, 국부 데이터는 프라이버시를 보호하도록 토큰화될 수 있다. 예를 들어, 사용자 전화 번호는 사용자들이 특정 제공자의 고객이고 우량한 자산 상태에 있지만 넘겨 주어지는 모두는 우량한 자산 상태 지위이고 사용자 이름 또는 전화 번호가 아니라고 말하도록 사용될 수 있다. 이는 제공자와 국부적으로 연락함으로써 행해지고 확인 데이터를 갖는 것은 원격으로 검증될 수 있는 제공자 트랜잭션 신원을 포함한다.

[0109] 일부 시스템은 국부 입증 데이터가 트랜잭션 시에 알려진 조건으로 있다는 것을 격리된 실행 환경이 입증할 수 있는 것을 보장하도록 국부 입증 데이터를 레버리징할 수 있다.

[0110] 시스템들은 특정 트랜잭션에 필요한 정책을 제공하도록 암호로 보장되는 로직 스크립트로 구성될 수 있다. 스크립트 입증은 트랜잭션 검증 데이터의 일부로서 포함될 수 있다.

[0111] 시스템들은 트랜잭션이 해제되기(즉, 클라이언트측 상의 다중 신호) 이전에 국부 또는 원격 승인들을 포함할 수 있다. 시스템들은 명령어가 예를 들어, 펌프의 속도를 증가시키는 실시간 상태에의 델타이도록 국부적으로 보장되고 그 다음 변경되는 실시간 데이터를 수신할 수 있다. 일부 시스템에서, 검증하는 디바이스는 트랜잭션이 최소 수의 파라미터를 충족시키는 알려진 소스에서 비롯되었다는 것을 보장한다. 다른 시스템들에서, 수신하는 디바이스는 게다가 국부 또는 원격 정보를 검증한다.

[0112] 본 발명이 특히 본 발명의 예시적 실시예들을 참조하여 나타내어지고 설명되었지만, 형태 및 세부 사항들의 다양한 변경이 첨부된 청구항들에 의해 포함되는 본 발명의 범위로부터 벗어나지 않는 범위 내에서 예시적 실시예들에 행해질 수 있다는 점이 당업자에 의해 이해될 것이다.

[0113] 부록

[0114] 1. 구성 요소 사양

[0115] · 구성 요소 사양

[0116] · 시스템 개요

[0117] · 원칙

[0118] · 시스템 구성 요소들

[0119] · 시스템 기능들

[0120] 2. 시스템 개요

[0121] Rivetz는 단순한 API를 통해 엔드포인트 디바이스들에서 웹 개발자들 및 앱 개발자들이 견고해진 암호화 및 신원 키들을 이용하는 것을 가능하게 한다. 이러한 시스템을 지원하기 위해, 본 발명은 신원 키들의 등록, 및 입증, 백업 및 디바이스 그룹화를 위한 디바이스 관리 서비스들의 세트를 관리한다.

[0122] Rivetz는 이하로 구성된다:

[0123] · 디바이스 하드웨어로 구현되는 소수의 프라이버시, 신원 및 인증 기능을 드러내는 클라이언트 모듈

[0124] · 디바이스들 및 서비스들의 등록 및 페어링을 가능하게 하는 Rivetz.net에서 호스팅되는 웹 서비스

[0125] · 명령어들이 서비스 제공자로부터 디바이스로 통신되는 프로토콜

[0126] Rivetz.net은 디바이스 관리, 백업, 입증 등을 위한 이러한 프레임워크에 의거한 서비스들을 추가로 제공할 것이다.

[0127] Rivetz.net은 디바이스들 및 서비스 제공자들의 신원 키들을 등록시키기 위해 Rivetz 개인 키를 사용하는 Python으로 기록된 JSON API이다. 등록 동안, 디바이스 또는 서비스 제공자의 공개 키는 Rivetz에 의해 기록된다. 등록은 Rivetz가 디바이스를 서비스 제공자와 페어링하는 것을 가능하게 한다. 페어링의 결과는 디바이스가 Rivetz에 의해 보증되는 서비스 공개 키를 갖고 그러므로 서비스 제공자 명령어들에 응답할 수 있는 것이다.

[0128] Rivetz 프로토콜은 명령어를 수락하도록 디바이스에 적용되어야 하는 서명/암호화 및 명령어의 구조체를 지정한다. 명령어 그 자체는 명령어 코드, 버전 데이터 및 페이로드를 포함하는 C 구조체로서 마련된다. 전체 구조체는 서비스 제공자 키에 의해 서명되고 디바이스 국부 커맨드를 호출함으로써 Rivet에 전달된다.

[0129] Rivetz는 모든 리벳화된 디바이스와의 영속적 연결을 유지하기 위해 보안적 소켓을 사용한다. 이러한 채널은 페어링 및 다른 관리상의 기능들에 사용된다.

[0130] Rivetz는 명령어의 구성 및 서명을 단순화하기 위해 서비스 제공자들에게 라이브러리 코드를 제공한다. 이러한 라이브러리는 Python으로 초기에 제공될 것이다. 다른 언어들이 뒤따를 것이다.

[0131] 3. 원칙

[0132] · 본 발명은 툴들을 웹 커뮤니티에 제공한다 - 본 발명의 고객들은 신뢰 가능한 디바이스 승인 및 실 암호화를 필요로 하는 방대한 수의 웹 서비스 및 앱이다. 아주 크게, 이러한 커뮤니티는 "서명" 및 "암호화"를 이해하고 방법을 지정하도록 요청될 때를 잃게 된다. 본 발명은 그것들에 대해 결정할 것이다.

[0133] · 본 발명은 실패점일 수 없다 - Rivetz는 사람이 사람의 신뢰를 전하는 다른 시스템일 수 없다. 본 발명은 등록, 페어링 및 관리 서비스들 (및 Rivet 그 자체)에서 소중한 역할을 하지만, 본 발명의 서버는 모든 트랜잭션에 대해 의존되지 않아야 한다.

[0134] · 본 발명은 사용자들을 추적하지 않는다 - 본 발명의 시스템은 디바이스들을 관리하도록 설계된다. 본 발명은 디바이스들을 작동시키는 사용자들을 식별하거나 추적하지 않는다.

[0135] · 본 발명은 하드웨어만을 신뢰한다 - Rivetz는 하드웨어에 의해 후원되는 암호화 프리미티브들만을 신뢰한다. 이용 가능하지 않을 때, 본 발명은 취약한 루트를 "견고하게 하도록" 시도하지 않을 것이고, 오히려 엔드포인트의 신뢰 레벨에 대해 선행할 것이다.

[0136] 4. 시스템 구성 요소들

[0137] 본 문서는 본 발명의 시스템을 포함하는 별개의 구성 요소들로 분할된다. 각각의 구성 요소의 경우, 본 발명은 각각의 구성 요소가 드러내는 기능들, 각각의 구성 요소가 관리하는 데이터 및 각각의 구성 요소의 현실화 배후의 구현 결정들을 설명한다.

[0138] 어떤 미션 크리티컬 데이터도 유지 관리하지 않고, 오히려 서비스 제공자들과 디바이스들 사이의 한결같은 한층 더 매우 보안적 연결들을 플랫폼에 제공하는 것이 Rivetz의 의도이다. 한편은 디바이스에 대한 명령어를 마련하는 Rivetz 인코더이고, 다른 한편은 그러한 명령어에 따라 작동할 수 있는 TEE 애플릿인 디바이스 Rivet이다. Rivetz 프로토콜은 이러한 명령어들 및 답신들이 구성되는 방법을 정의한다.

[0139] 새로운 구성 요소의 제목: 

구성 요소	정의
디바이스 Rivet	물리적인 작업과 디지털 작업 사이의 본 발명의 결속을 구현하는 Rivetz TEE 애플릿. 디바이스 Rivet은 하드웨어에 신원, 트랜잭션 및 입증의 특징부들을 락킹하고 본 발명의 기술적 제안의 기반을 형성한다.
링 관리자	링 관리자는 디바이스들의 컬렉션들 (또는 링들)을 관리하기 위해 최종 사용자들에게 제공되는 서비스이다. 디바이스들은 단일 신원으로 그룹화되고 서로 백업하고 보증하는데 사용될 수 있다. 링들은 디바이스들의 네트워크를 생성하기 위해 다른 링들과 연관될 수 있다.
Rivet 어댑터	Rivet 어댑터는 TEE로 개재되는 디바이스 Rivet과 파트너 앱들 및 온라인 서비스들의 외부 세계 사이의 인터페이스이다. 구현에서, Rivet 어댑터는 하나 이상의 다양한 형태로 나타난다. 본 발명이 디바이스들, 하드웨어 지원 및 OS 아키텍처에 걸친 동일한 기본 능력들을 제공하도록 노력하지만, 본 발명은 무엇이 실제로 가능한지 그리고 이러한 특징들이 제공되는 방법을 결정할 것이다.
Rivetz 인코더	Rivetz 인코더는 명령어 기록을 생성하고 응답 기록을 처리한다. 이들은 디바이스 Rivet(trustlet)에 정의되고, 이것에 의해 해석되는 메시지 데이터 구조체들이다.
RivetzNet	RivetzNet은 디바이스들 및 서비스 제공자들을 보증된 관계로 페어링하는 Rivetz에 의해 작동되는 서비스이다.

[0140]

[0141] 5. 시스템 기능들

[0142] Rivetz 사용 사례들을 참조하시오.

[0143] 6. 링 관리자

[0144] 링 관리자는 디바이스들의 컬렉션들 (또는 링들)을 관리하기 위해 최종 사용자들에게 제공되는 서비스이다. 디바이스들은 단일 신원으로 그룹화되고 서로 백업하고 보증하는데 사용될 수 있다. 링들은 디바이스들의 네트워크를 생성하기 위해 다른 링들과 연관될 수 있다.

[0145] · 링 관리자

[0146] · 구성 요소 콘텍스트

[0147] · 구성 요소 도해

[0148] · 구성 요소 분해

[0149] · 엔티티 책임

[0150] · 인터페이스 사양

[0151] 7. 구성 요소 콘텍스트

[0152] (패키지, 패턴들, 프레임워크들, 전제 조건들, 사용량)

[0153] 8. 구성 요소 도해

[0154] 9. 구성 요소 분해

[0155] 새로운 구성 요소의 제목: Submit

[0156]

구성 요소	정의

[0157] 10. 엔티티 책임

[0158] (이러한 구성 요소에 의해 제어되는 사업 또는 기술적 엔티티들)

[0159] 11. 인터페이스 사양

[0160] 12. RivetzNet

[0161] RivetzNet은 디바이스들 및 서비스 제공자들을 보증된 관계로 페어링하는 Rivetz에 의해 작동되는 서비스이다.

[0162] 본래, 본 발명은 영구성 및 투명성을 위해 디바이스 등록을 네임코인으로 표현하도록 의도하였지만, 프라이버시 우려들은 이러한 계획을 당분간 보류하였다. 본 발명이 디바이스들 상의 입증 데이터를 수집하기 시작함에 따라, 이러한 결정이 재평가될 것이다. (세부 사항에 대한 논제 이력 참조).

[0163] · RivetzNet

[0164] · 구성 요소 콘텍스트

[0165] · 웹 API

[0166] · 개인 키

[0167] · 엔티티 책임

[0168] · 인터페이스 사양

[0169] · 디바이스 등록

[0170] · 서비스 제공자 등록

[0171] · 디바이스 ID를 얻음

[0172] · 디바이스를 페어링함

[0173] · 사용 사례 참조

[0174] 13. 구성 요소 콘텍스트

[0175] RivetzNet은 디바이스에 등록되는 제1 서비스 제공자이고 부가 서비스 제공자들을 그러한 디바이스와 페어링할 수 있는 특별한 능력을 갖는다.

[0176] 14. 웹 API

[0177] 웹 API와의 모든 통신이 승인될 필요가 있다. 본 발명은 API 키 아니면, SSL 키 스왑을 사용할 수 있다. 본 발명은 모든 요청이 서명될 것을 요청할 수 있거나, 본 발명은 본 발명의 시스템을 사용하기에 단순하게 유지하는 것을 인식하고 있어야 한다.

[0178] 15. 개인 키

[0179] 디바이스들과의 Rivetz 관계는 본 발명의 개인 키로 명령어들에 서명할 수 있는 것에 의존한다. 본 발명이 이러한 키를 보호하는 것이 물론 가장 중요하다. 본 발명은 키를 HSM에 넣도록 요구해야 한다.

[0180] 16. 엔티티 책임

[0181] (이러한 구성 요소에 의해 제어되는 사업 또는 기술적 엔티티들)

[0182] 새로운 엔티티의 제목: Submit

엔티티	정의
디바이스 ID	RivetzNet 또는 다른 등록 에이전트에 의해 디바이스에 할당되는 UUID로의 고유 식별자.
디바이스 포인터	임의의 국부 애플리케이션에 의해 요청될 수 있는 디바이스에 대한 단명하는 포인터. 디바이스 포인터는 RivetzNet에 대한 현재의 소켓 세션을 식별할 수 있고 그러므로 디바이스 통신 채널을 확립하고 영구적 식별자인, 디바이스 ID를 검색하는데 사용될 수 있다.
디바이스 등록 기록	디바이스 등록의 루트는 고유한 익명의 식별자, 등록 날짜, 디바이스 하드웨어에 유지되는 개인 키와 페어링된 공개 키 및 등록 에이전트(현재 Rivetz인 것으로 가정됨)로부터의 보증 시그니처를 포함한다.
디스패치 ID	RivetzNet으로부터 Rivet 어댑터에 의해 복귀되는 응답 기록으로 송신되는 명령어 기록에 부합시키는데 사용되는 고유 식별자.
Rivetz 코인 계정	RivetzNet은 RivetzNet의 등록들을 저장하고, 스탬핑하고, 공개하기 위해 블록 체인 인프라 구조체(현재 네임코인)를 사용한다. 이는 블록 체인에서의 이름/값 쌍 기록을 구매함으로써 작동하고 따라서 발생 계정을 가져야 한다. Rivetz 제어 계정이 기록을 구매하였다는 사실은 보증으로 해석된다.
Rivetz 신원 키	Rivetz Corp.의 보증을 나타내도록 생성되는 고유 공개/개인 키 쌍. 이러한 키 쌍은 흔히 회전되고 하드웨어에서 보호될 것이다. 이상적으로, 본 발명의 프로토콜들은 키 쌍이 절도되더라도 시스템이 지나치게 위해되지 않는 정도일 것이다.
서비스 제공자 ID	RivetzNet에 의해 서비스 제공자에 할당되는 고유 식별자.
서비스 제공자 등록 기록	명령어들을 리벳화된 디바이스로 송신하기를 원하는 각각의 등록된 서비스 제공자에 대해 생성되는 기록. 이는 서비스 제공자 이름, 등록 날짜, 공개 키 및 (Rivetz에 의한) 보증 시그니처를 포함한다.

[0183]

[0184] 17. 인터페이스 사양

[0185] 18. 디바이스 등록

[0186] 고유 식별자 및 공개 키가 블록 체인에서 이러한 결속의 기록을 구매한다고 가정한다. 구매는 Rivetz 코인 계정으로 행해지므로, 등록을 보증한다. 이상적으로, Rivetz 시그니처는 디바이스가 OEM으로부터의 보증 키를 공급할 수 있으면, 단지 적용될 것이다.

[0187] 19. 서비스 제공자 등록

[0188] 주어진 체계에 대한 서비스 제공자 ID를 생성한다. 등록은 또한 SP가 통신을 검증하기 위해 Rivetz 인코더의 URL의 구현 및 공개 신원을 호스팅하는 URL을 포함해야 한다.

[0189] 20. 디바이스 ID를 얻음

[0190] 디바이스 포인터가 알려진 디바이스 ID를 요청하는 서비스 제공자로 복귀시킨다고 가정한다.

논의	정의
서비스 제공자 ID	RivetzNet에 의해 서비스 제공자에 할당되는 고유 식별자.
디바이스 포인터	임의의 국부 애플리케이션에 의해 요청될 수 있는 디바이스에 대한 단명하는 포인터. 디바이스 포인터는 RivetzNet에 대한 현재의 소켓 세션을 식별할 수 있고 그러므로 디바이스 통신 채널을 확립하고 영구적 식별자인, 디바이스 ID를 검색하는데 사용될 수 있다.

[0191]

[0192] 복귀: 디바이스 ID

[0193] 21. 디바이스를 페어링함

[0194] 서비스 제공자가 명령어를 송신할 수 있기 전에, 서비스 제공자는 타겟 디바이스에 서비스 제공자의 id 및 공개키를 등록해야 한다. 이는 명령어를 실행시키기 전에 명령어의 근원을 디바이스가 확인하는 것을 가능하게 한다. 디바이스를 페어링하는 것은 디바이스 상에 새로운 신원 키를 자동적으로 생성할 것이다.

논의	정의
서비스 제공자 ID	RivetzNet에 의해 서비스 제공자에 할당되는 고유 식별자.
디바이스 포인터	임의의 국부 애플리케이션에 의해 요청될 수 있는 디바이스에 대한 단명하는 포인터. 디바이스 포인터는 RivetzNet에 대한 현재의 소켓 세션을 식별할 수 있고 그러므로 디바이스 통신 채널을 확립하고 영구적 식별자인, 디바이스 ID를 검색하는데 사용될 수 있다.

[0195]

[0196] 22. 사용 사례 참조

[0197] · Rivetz에의 디바이스 등록 - Rivet이 임의의 것을 할 수 있기 전에, Rivet은 RivetzNet에 등록할 필요가 있다. 등록은 고유 신원 키의 생성을 야기한다. 등록은 보증...에 의존한다.

[0198] · 서비스 제공자에의 디바이스 등록 - 서비스 제공자는 디바이스가 임의의 요청들에 응답하기 전에, 그러한 디바이스에 등록되는 서비스 제공자의 서비스 제공자 ID 및 공개 신원 키를 가질 필요가 있다. 임의의 경우에도...

[0199] · Rivetz에의 서비스 제공자 등록 - RivetzNet (http://rivetz...상의 형태를 채우는 것으로서 서비스 제공자 초기 등록이 단순함에 따라, Rivetz 시스템으로 코딩할 것을 요구하는 누구나 등록할 필요가 있다.

[0200] 웹 홈 > 두문자어 표 > HSM

[0201] 하드웨어 보안 모듈은 강한 승인을 위해 디지털 키들을 보호하고 관리하고 암호화 처리를 제공하는 물리적 컴퓨팅 디바이스이다.

[0202] 1. 디바이스 ID

[0203] RivetzNet 또는 다른 등록 에이전트에 의해 디바이스에 할당되는 UUID로의 고유 식별자.

[0204] 2. 디바이스 포인터

[0205] 임의의 국부 애플리케이션에 의해 요청될 수 있는 디바이스에 대한 단명하는 포인터. 디바이스 포인터는 RivetzNet에 대한 현재의 소켓 세션을 식별할 수 있고 그러므로 디바이스 통신 채널을 확립하고 영구적 식별자인, 디바이스 ID를 검색하는데 사용될 수 있다.

[0206] 데이터 타입:

[0207] 3. Rivetz 신원 키

[0208] Rivetz Corp.의 보증을 나타내도록 생성되는 고유 공개/개인 키 쌍. 이러한 키 쌍은 흔히 회전되고 하드웨어에서 보호될 것이다. 이상적으로, 본 발명의 프로토콜들은 키 쌍이 절도되더라도 시스템이 지나치게 위해되지 않는 정도일 것이다.

[0209] 4. 디바이스 등록 기록

[0210] 디바이스 등록의 루트는 고유한 익명의 식별자, 등록 날짜, 디바이스 하드웨어에 유지되는 개인 키와 페어링된 공개 키 및 등록 에이전트(현재 Rivetz인 것으로 가정됨)로부터의 보증 시그니처를 포함한다.

[0211] 5. 디스패치 ID

[0212] RivetzNet으로부터 Rivet 어댑터에 의해 복귀되는 응답 기록으로 송신되는 명령어 기록에 부합시키는데 사용되는 고유 식별자.

[0213] 6. Rivetz 코인 계정

[0214] RivetzNet은 RivetzNet의 등록들을 저장하고, 스탬핑하고, 공개하기 위해 블록 체인 인프라 구조체(현재 네임코인)를 사용한다. 이는 블록 체인에서의 이름/값 쌍 기록을 구매함으로써 작동하고 따라서 발생 계정을 가져야 한다. Rivetz 제어 계정이 기록을 구매하였다는 사실은 보증으로 해석된다.

[0215] 7. 서비스 제공자 ID

[0216] RivetzNet에 의해 서비스 제공자에 할당되는 고유 식별자.

[0217] 8. 서비스 제공자 등록 기록

[0218] 명령어들을 리벳화된 디바이스로 송신하기를 원하는 각각의 등록된 서비스 제공자에 대해 생성되는 기록. 이는 서비스 제공자 이름, 등록 날짜, 공개 키 및 (Rivetz에 의한) 보증 시그니처를 포함한다.

[0219] 9. Rivetz 인코더

[0220] Rivetz 인코더는 명령어 기록을 생성하고 응답 기록을 처리한다. 이들은 디바이스 Rivet(trustlet)에 정의되고, 이것에 의해 해석되는 메시지 데이터 구조체들이다.

[0221] a. 구성 요소 콘텍스트

[0222] Rivetz 인코더는 본 발명의 파트너들에 의해 호스팅되도록 기록되는 소프트웨어이다.

[0223] Rivetz 인코더는 공개 개방 소스로서 분배된다.

[0224] b. 엔티티 책임

[0225] 새로운 엔티티의 제목: 

[0226] c. 인터페이스 사양

[0227] d. 구현

[0228] e. 사용 사례 참조

[0229] 어떤 것을 암호화함 - Rivetz는 텍스트 또는 이미지들을 암호화하는 메커닉들을 제공하지만 파트너들의 서비스에 대한 인터페이스가 메시징 애플리케이션이든, 파트너들이 파트너들의 서비스에 대한 인터페이스를 기획하는 것을 기대한다.

[0230] 10. 서비스 제공자 신원 키

[0231] 서비스 제공자 신원의 사적 부분은 명령어들에 서명하기 위해 Rivetz 인코더에 의해 사용된다. 공적 부분은 Rivetz 및 페어링된 디바이스들에 제공된다.

[0232] 11. 디바이스 Rivet

[0233] 물리적인 작업과 디지털 작업 사이의 본 발명의 결속을 구현하는 Rivetz TEE 애플릿. 디바이스 Rivet은 하드웨

어에 신원, 트랜잭션 및 입증의 특징부들을 락킹하고 본 발명의 기술적 제안의 기반을 형성한다.

[0234] · 디바이스 Rivet

[0235] · 구성 요소 콘텍스트

[0236] · 구성 요소 설명

[0237] · 엔티티 책임

[0238] · 인터페이스 사양

[0239] · 디바이스를 등록함

[0240] · 키를 생성함

[0241] · 키로 암호화함

[0242] · 키로 복호화함

[0243] · 명령어를 처리함

[0244] · 사용 사례 참조

[0245] · 비고

[0246] a. 구성 요소 콘텍스트

[0247] 본 발명은 디바이스 Rivet 구현을 호스팅하는 2개의 타겟 플랫폼: 안드로이드 상의 Trustonic 및 Windows PC들에 대한 Intel ME를 현재 갖는다. 환경들 둘 다는 제한된 처리를 갖고 상세하게는 보안 및 리소스 사용을 위해 단순하도록 설계된다.

[0248] Trustonic 신뢰 앱들(TA's)은 C에서의 안드로이드 NDK 컴파일러로 구현된다. TA와 인터페이싱하는 것은 공유된 메모리 버퍼를 사용하여 행해진다. 커맨드들은 메모리 블록으로 패키징되고 통지는 TA를 로딩하고 실행시키기 위해 Trustonic 제어기로 송신된다. 통지는 동기적이다. 호스트 앱(정기적 안드로이드 앱)은 응답을 대기한다. 신뢰된 앱은 호스트 상에 신뢰된 앱의 데이터를 저장하는 것으로 기대되지만, Trustonic 제어기는 TEE에서 실행될 때, 데이터가 개방될 수만 있도록 보안적 래퍼를 제공한다.

[0249] Intel 구현의 경우, 앱들은 Java에 기록되고 Intel의 마스터 키에 의해 서명된다. 본 발명은 이를 위해 Intel로부터 DAL SDK를 얻을 수 있었고 Intel은 12 월에 시작하여 본 발명의 노고로부터의 적극적인 지원을 나타내었다.

[0250] b. 구성 요소 설명

[0251] 구현은 플랫폼들에 걸쳐 매우 상이하고 Rivet 어댑터와의 통합은 디바이스 특정 방법들을 추가로 발생시킬 것이다. 그러나, 논리적 구현은 동일한 것으로 의도되고 입력 데이터 구조체들은 필연적으로 동일하다. Rivetz 시스템의 나머지는 모두가 동일한 인터페이스를 지원하지만, 일부가 더 많거나 더 적은 특징 세트를 갖는 것과 같은 디바이스들을 처리하기를 원한다.

[0252] 디바이스 Rivet(Trustlet)에서 이하의 기능성의 3개의 주영역이 있다:

[0253] · 디바이스 등록 - 이는 디바이스 Rivet이 등록 에이전트(RivetzNet)로 신원을 확립하는 프로세스이다.

[0254] · 명령어 처리 - 주어진 명령어를 실행시킨다. 이는 서비스 제공자에서 비롯되는 서명된 데이터 구조체이다.

[0255] · 보안 프리미티브들 - 국부 애플리케이션 사용에 대해 노출되는 단순한 보안 기능성.

[0256] c. 엔티티 책임

[0257] 새로운 엔티티의 제목: Submit

엔티티	정의
계정 키들	계정 키들은 디바이스 Rivet에 의해 보안적으로 유지된다. 계정 키들은 신뢰된 실행 환경의 한계들을 결코 떠나지 않는다. 계정 키들은 디바이스에 결속되는 보안적 래퍼에서 생성되고, 저장되고, 적용된다.
계정 핀	계정 키들은 계정 키들이 임의의 트랜잭션에서 적용되기 전에, 사용자 동의를 테스트하는데 사용되는 계정 핀에 결속될 수 있다.
명령어 페이로드	디바이스 Rivet으로의 명령어 기록에 의해 전해지는 데이터 블랍. 명령어 페이로드는 명령어 타입에 따라 해석된다.
명령어 기록	Rivetz 명령어는 식별된 디바이스 Rivet에 의해 처리되도록 타겟화되는 데이터 패키지이다. Rivetz 명령어는 Rivetz TEE 애플릿에서 일부 작동을 수행하도록 디바이스를 명령하기 위해 커맨드, 페이로드 및 필요한 시그니처들을 포함한다.
명령어 시그니처	디바이스 Rivet에 대해 예정되는 모든 명령어는 발하는 서비스 제공자에 의해 서명되어야 한다. 서비스 제공자는 RivetzNet에 등록했어야 한다. 등록된 서비스 제공자는 Rivetz에 의해 보증되고 모든 등록된 디바이스에 분배되는 등록된 서비스 제공자의 공개 키를 가질 것이다.
응답 기록	복귀 상태 및 명령어 기록의 처리에 기인하는 페이로드.

[0258]

[0259] d. 인터페이스 사양

[0260] i. 디바이스를 등록함

[0261] ii. 키를 생성함

[0262] iii. 키로 암호화함

[0263] TEE 어댑터는 서비스 제공자 기록에서의 명명된 암호화 키를 검색한다.

[0264] iv. 키로 복호화함

[0265] v. 명령어를 처리함

[0266] e. 사용 사례 참조

[0267] · 키 생성 - 양 서명 및 암호화에 대한 디바이스 Rivet에서의 키 쌍을 생성한다. 행위자들 서비스 제공자 설명 Rivetz의 주목적은 보안하고 적용하는 것...이다.

[0268] · 국부 사용자 생성 - 어떤 서비스 제공자 인증도 행위자들인, 제품 행위자들로부터의 선택/생성 행위자들... 이 주어지지 않는 경우들에서 Rivet의 사용을 인증할 수 있는 국부 엔티티를 확립한다.

[0269] · 어떤 것을 암호화함 - Rivetz는 텍스트 또는 이미지들을 암호화하는 메커닉들을 제공하지만 파트너들의 서비스에 대한 인터페이스가 메시징 애플리케이션이든..., 파트너들이 파트너들의 서비스에 대한 인터페이스를 기획하는 것을 기대한다.

[0270] · Rivetz에의 디바이스 등록 - Rivet이 임의의 것을 할 수 있기 전에, Rivet은 RivetzNet에 등록할 필요가 있다. 등록은 고유 신원 키의 생성을 야기한다. 등록은 보증...에 의존한다.

[0271] 12. 명령어 페이로드

[0272] 디바이스 Rivet으로의 명령어 기록에 의해 전해지는 데이터 블랍. 명령어 페이로드는 명령어 타입에 따라 해석

된다.

[0273] 13. 명령어 기록

[0274] Rivetz 명령어는 식별된 디바이스 Rivet에 의해 처리되도록 타겟화되는 데이터 패키지이다. Rivetz 명령어는 Rivetz TEE 애플릿에서 일부 작동을 수행하도록 디바이스를 명령하기 위해 커맨드, 페이로드 및 필요한 시그니처들을 포함한다.

[0275] 대부분의 명령어는 응답 기록의 구성 및 복귀를 야기할 것이다. 응답 기록은 Rivetz 디스패치에 의해 서비스 제공자로 다시 전달될 것이다.

[0276] a. 데이터 구조체

파라미터	타입/크기	설명
버전 ID	정수	호환성을 위한 데이터 구조체에 대한 버전 id 타입.
서비스 제공자 ID	UUID	이러한 명령어를 발하는 서비스 제공자의 고유 식별자.
명령어 타입	정수	명령어 타입 식별자. 이는 페이로드의 내용들을 해석하는 방법을 결정한다.
명령어 페이로드	블랍	임의적인 데이터 블랍.
명령어 시그니처	바이트 (512)	서비스 제공자 키에 의해 서명되는 명령어의 해시.

[0277]

[0278] b. 명령어 타입들

타입 명칭	값	설명
RIVETZ_텍스트_확인_행함		페이로드는 텍스트 메시지 및 서명된 해시를 포함한다. 메시지 스트링은 확인 및 취소 버튼과 함께 표시될 것이다. 확인 시에, 디바이스는 메시지에 서명하고 메시시를 복귀시킬 것이다.
RIVETZ_이미지_확인_행함		페이로드는 이미지 및 서명된 해시를 포함한다. 이미지는 확인 및 취소 버튼들과 함께 표시될 것이다.
RIVETZ_이미지_표시		페이로드는 디바이스 키로 암호화되는 이미지 및 발행자에 의해 서명되는 해시를 포함한다. 이미지는 디바이스 Rivet에 의해 표시된다. 어떤 복귀도 없다.
RIVETZ_텍스트_표시		페이로드는 디바이스 키로 암호화되는 텍스트 및 발행자에 의해 서명되는 해시를 포함한다. 텍스트는 디바이스 Rivet에 의해 렌더링된다. 어떤 복귀도 없다.
RIVETZ_비트코인_계정_생성		새로운 비트코인 계정이 생성되고 공개 어드레스가 복귀된다.
RIVETZ_SP_목록_업데이트		페이로드는 등록 에이전트(이는 Rivetz임)에 의해 등록되었던 서비스 제공자들의 ID들 및 공개 키들을 포함한다. 이러한 목록은 디바이스를 등록했던 등록 에이전트에 의해 서명된다. 즉, 디바이스를 등록했던 시스템만이 등록된 서비스 제공자들의 목록을 업데이트할 수 있다.
RIVETZ_VC_TXN_서명	0x0001	페이로드는 디바이스 Rivet에 의해 유지 관리되는 명명된 비트코인 계정 키에 서명되게 될 충분히 상주된 가상 코인(비트코인, 라이트코인, 피어코인 등) 트랜잭션을 포함한다.
RIVETZ_키_추가	0x0101	페이로드는 기존 키를 서비스 제공자 키 목록에 추가하는 데이터를 포함한다. 새로운 키가 정상적 세계에서 결코 보이지 않도록 새로운 키를 생성할 것을 권하였다.
RIVETZ_키_얻음	0x0102	페이로드는 키 기록으로부터 공개 키를 회수하라

[0279]

		는 요청을 포함한다.
RIVETZ_키_삭제	0x0103	페이로드는 키 기록을 삭제하라는 요청을 포함한다.
RIVETZ_ENUM_키	0x0104	페이로드는 키 기록들의 목록을 얻으라는 요청을 포함한다.
RIVETZ_ECDSA_생성	0x0201	페이로드는 ECDSA 공개 및 개인 키들을 생성하라는 요청을 포함한다. 키는 Rivet 안드로이드에서의 키 기록에 저장된다.
RIVETZ_ECDSA_서명	0x0202	페이로드는 ECDSA 개인 키를 사용하여 데이터에 서명하라는 요청을 포함한다.
RIVETZ_ECDSA_검증	0x0203	페이로드는 ECDSA 공개 키를 사용하여 데이터를 검증하라는 요청을 포함한다.
RIVETZ_ECDSA_PUBPRV를 얻음	0x0204	페이로드는 ECDSA 개인 키로부터 공개 가상 코인(비트코인, 라이트코인, 피어코인 등) 어드레스를 얻으라는 요청을 포함한다.
RIVETZ_ECDSA_PUBSIG를 얻음	0x0205	페이로드는 시그니처 및 메시지로부터 ECDSA 공개 키를 얻으라는 요청을 포함한다.
RIVETZ_ECDH_암호화	0x0301	페이로드는 ECDH를 사용하여 데이터를 암호화하라는 요청을 포함한다.
RIVETZ_ECDH_복호화	0x0302	페이로드는 ECDH를 사용하여 데이터를 복호화하라는 요청을 포함한다.

[0280]

[0281] 모든 디바이스가 모든 명령어를 지원할 수 있을 것은 아니라는 것을 주목해야 한다. 명령어가 지원되지 않으면, 디바이스 Rivet은 지원되지_않음을 복귀시킬 것이다. 응답 기록 참조.

[0282] 14. 명령어 타입

[0283] 명령어 기록의 타입을 나타내는 상수값. 이는 명령어 페이로드가 해석되게 될 방법을 결정한다.

[0284] 명령어 타입들을 명령어 기록에 기술한다.

[0285] 15. 명령어 시그니처

[0286] 디바이스 Rivet에 대해 예정되는 모든 명령어는 발하는 서비스 제공자에 의해 서명되어야 한다. 서비스 제공자는 RivetzNet에 등록했어야 한다. 등록된 서비스 제공자는 Rivetz에 의해 보증되고 모든 등록된 디바이스에 분배되는 등록된 서비스 제공자의 공개 키를 가질 것이다.

[0287] 16. 계정 키들

[0288] 계정 키들은 디바이스 Rivet에 의해 보안적으로 유지된다. 계정 키들은 신뢰된 실행 환경의 한계들을 결코 떠나지 않는다. 계정 키들은 디바이스에 결속되는 보안적 래퍼에서 생성되고, 저장되고, 적용된다.

[0289] 17. 계정 핀

[0290] 계정 키들은 계정 키들이 임의의 트랜잭션에서 적용되기 전에, 사용자 동의를 테스트하는데 사용되는 계정 핀에 결속될 수 있다.

[0291] 18. 응답 기록

[0292] 복귀 상태 및 명령어 기록의 처리에 기인하는 페이로드.

[0293] a. 상태 코드들

복귀 코드 명칭	설명
복귀_명령어_실행됨	디바이스 Rivet에 의해 실행되었던 명령어에 대한 포괄적 복귀.
복귀_지원되지_않음	명령어 기록에서 제공되는 명령어 타입은 이러한 디바이스 상에 지원되지 않는다.
복귀_알려지지_않음	명령어 기록에서 제공되는 명령어 타입은 알려지지 않는다.
복귀_확인_정상	확인에 대한 요청이 사용자에 의해 확인되었다. 복귀의 페이로드는 디바이스에 의해 서명되는 확인 객체(이미지 또는 텍스트)의 해시를 포함할 것이다.
복귀_확인_취소됨	확인에 대한 요청이 사용자에 의해 취소되었다.
복귀_확인_만료됨	확인에 대한 요청이 제한 시간 내에 사용자에 의해 확인되지도 않고 취소되지도 않았다.

[0294]

[0295] 19. Rivet 어댑터

[0296] Rivet 어댑터는 TEE로 개재되는 디바이스 Rivet과 파트너 앱들 및 온라인 서비스들의 외부 세계 사이의 인터페이스이다. 구현에서, Rivet 어댑터는 하나 이상의 다양한 형태로 나타난다. 본 발명이 디바이스들, 하드웨어 지원 및 OS 아키텍처에 걸친 동일한 기본 능력들을 제공하도록 노력하지만, 본 발명은 무엇이 실제로 가능한지 그리고 이러한 특징들이 제공되는 방법을 결정할 것이다.

[0297] · Rivet 어댑터

[0298] · 도해

[0299] · 하위 구성 요소들

[0300] · 구현

[0301] · 사용 사례 참조

[0302] a. 도해

[0303] b. 하위 구성 요소들

[0304] Rivet 어댑터는 밖으로 그리고 안으로 본 인터페이스들로 구성된다. 안으로 본 인터페이스인, TEE 어댑터는 trustlet(디바이스 Rivet)과의 전매 통신을 처리한다. 호스트 어댑터는 제3 자 애플리케이션들에 서비스들을 노출시키도록 제공된다.

[0305] 인터페이스 및 구현 세부 사항들에 대한 개별 하위 구성 요소들을 참조하시오.

[0306] 호스트 어댑터 -- 호스트 어댑터는 브라우저들 또는 시스템 서비스들과 같은 상이한 국부 콘텍스트들을 통해 Rivet 어댑터의 인터페이스를 제공한다. 초기에 이는 안드로이드 서비스 및 Windows 기업 프로세스이지만, 다양한 콘텍스트에 대한 다수의 실현이 예상된다.

[0307] 소켓 어댑터 -- 클라이언트 환경을 RivetzNet에 연결한다.

[0308] TEE 어댑터 -- 이러한 구성 요소는 Trustonic 또는 Intel ME에서 실행되는 본 발명의 trustlet으로 커맨드들을 송신하는 전매 글루이다.

[0309] c. 구현

[0310] 안드로이드 구현에서, Rivet 어댑터는 안드로이드 NDK 서비스 앱으로서 나타난다. Rivet 어댑터는 부팅에서 런칭하도록 구성된다. Rivet 어댑터는 Trustlet으로 송신되는 메시지 버퍼들을 마련하고 그 다음 응답 이벤트의

통지를 동기적으로 대기한다. 안드로이드 앱의 출현은 제3 자가 트리거하는 일련의 의도들을 제공한다. 앱, NDK 2진수들 및 Trustlet은 모두 분배를 위해 단일 APK로 패키징된다.

[0311] d. 사용 사례 참조

[0312] · 국부 사용자 생성 - 어떤 서비스 제공자 인증도 행위자들인, 제품 행위자들로부터의 선택/생성 행위자들... 이 주어지지 않는 경우들에서 Rivet의 사용을 인증할 수 있는 국부 엔티티를 확립한다.

[0313] · 어떤 것을 암호화함 - Rivetz는 텍스트 또는 이미지들을 암호화하는 메커닉들을 제공하지만 파트너들의 서비스에 대한 인터페이스가 메시징 애플리케이션이든..., 파트너들이 파트너들의 서비스에 대한 인터페이스를 기획하는 것을 기대한다.

[0314] · Rivetz에의 디바이스 등록 - Rivet이 임의의 것을 할 수 있기 전에, Rivet은 RivetzNet에 등록할 필요가 있다. 등록은 고유 신원 키의 생성을 야기한다. 등록은 보증...에 의존한다.

[0315] · 서비스 제공자에의 디바이스 등록 - 서비스 제공자는 디바이스가 임의의 요청들에 응답하기 전에, 그러한 디바이스에 등록되는 서비스 제공자의 서비스 제공자 ID 및 공개 신원 키를 가질 필요가 있다. 임의의 경우에도...

[0316] 20. 호스트 어댑터

[0317] 호스트 어댑터는 브라우저들 또는 시스템 서비스들과 같은 상이한 국부 콘텍스트들을 통해 Rivet 어댑터의 인터페이스를 제공한다. 초기에 이는 안드로이드 서비스 및 Windows 기업 프로세스이지만, 다양한 콘텍스트에 대한 다수의 실현이 예상된다.

[0318] 호스트 어댑터는 주로 호스트 환경에서 TEE 어댑터를 격리시키는 곳에 있다. 그러나, 호스트 어댑터는 호스트 기계 상의 최소 UI 존재를 갖는다. 호스트 어댑터는 "어바웃(About)" 페이지를 제공하고 최종 사용자가 최종 사용자의 앱들 목록에서 식별할 수 있는 아이템이다.

[0319] 궁극적으로, 호스트 어댑터는 백업 또는 가입과 같은 링 관리자 서비스들을 제공할 것이다.

[0320] · 호스트 어댑터

[0321] · 인터페이스

[0322] · 포인터를 얻음

[0323] · 해시를 얻음

[0324] · 실행

[0325] · 암호화

[0326] · 복호화

[0327] · 안드로이드 구현

[0328] · 안드로이드 인텐트 문서화

[0329] · Windows 구현

[0330] · 사용 사례 참조

[0331] a. 인터페이스

[0332] 호스트 어댑터는 잠재적으로 반대하는 환경에서 작동한다. 그러므로, 본 발명은 전형적으로 클라이언트가 위해되지 않았다는 제한된 보장을 가질 것이다. 그러므로, 호스트 어댑터의 역할은 주로 디바이스 Rivet에 대한 쉬운 액세스를 용이하게 하는 것이다. 디바이스 Rivet에 대해 의도되는 서비스 제공자로부터의 명령어들은 서비스 제공자에 의해 서명되고 그 다음 실행 명령어를 통해 TEE 어댑터 및 디바이스 Rivet으로 패스될 것이다. 국부 서비스 제공자 역할을 이용하도록 의도되는 명령어들은 호스트 어댑터에 의해 구성되고 그 다음 명령어가 디바이스 Rivet으로 패스되기 이전에 TEE 어댑터 또는 다른 엔티티에 의해 서명될 수 있다.

[0333] 암호화 및 복호화와 같은 일정 국부 서비스들은 국부 서비스 제공자 역할을 이용하여 호출되는 것이 가능해지고 호스트 어댑터는 본 발명의 고객들의 편의를 위해 국부적으로 이러한 서비스들에 인터페이스를 제공한다. 이들

은 일정 플랫폼들 상에서 불가능해질 수 있다.

[0334] i. 포인터를 얻음

[0335] 본 발명은 남용으로부터 영구적 디바이스 식별자들을 보호하기를 원한다. 입증된 서비스 제공자는 "이것이 무슨 디바이스입니까"라고 물어볼 필요가 있을 것이다. 악성 앱이 동일한 질문으로 유용한 응답을 얻을 수 없도록, 본 발명은 디바이스 포인터를 사용한다. 디바이스 포인터는 RivetzNet와의 소켓 연결 동안만 유효한 식별자이다. 수중에 있는 디바이스 포인터로, 서비스 제공자는 영구적 디바이스 ID에 대해 직접 RivetzNet에 질의하거나 페어링을 요청할 수 있다. 소켓 어댑터는 소켓 어댑터가 RivetzNet에 연결될 때마다, 메모리에 디바이스 포인터를 저장한다.

논의
없음

[0336]

[0337] 복귀: 디바이스 포인터 -- 임의의 국부 애플리케이션에 의해 요청될 수 있는 디바이스에 대한 단명하는 포인터. 디바이스 포인터는 RivetzNet에 대한 현재의 소켓 세션을 식별할 수 있고 그러므로 디바이스 통신 채널을 확립하고 영구적 식별자인, 디바이스 ID를 검색하는데 사용될 수 있다.

[0338] ii. 해시를 얻음

[0339] 명령어들에 서명하고 이것들을 암호화하기 위해, 서비스 제공자는 객체의 해시에 서명할 필요가 있다.

논의	정의
데이터 블랍	임의의 길이의 바이트들의 지정되지 않은 컬렉션으로서의 데이터.

[0340]

[0341] 복귀: 서명된해시 -

[0342] iii. 실행

[0343] 명령어 기록을 TEE 어댑터로 패스하고 응답 기록을 복귀시킨다. Rivet는 Rivet가 위험을 벗어나 패스되는 서비스 제공자 ID를 필요로 하도록 명령어를 처리할 콘텍스트가 주어질 필요가 있을 것이다.

논의	정의
서비스 제공자 ID	RivetzNet에 의해 서비스 제공자에 할당되는 고유 식별자.
명령어 기록	Rivetz 명령어는 식별된 디바이스 Rivet에 의해 처리되도록 타겟화되는 데이터 패키지이다. Rivetz 명령어는 Rivetz TEE 애플릿에서 일부 작동을 수행하도록 디바이스를 명령하기 위해 커맨드, 페이로드 및 필요한 시그니처들을 포함한다.

[0344]

[0345] 복귀: 응답 기록 -- 복귀 상태 및 명령어 기록의 처리에 기인하는 페이로드.

[0346] iv. 암호화

논의	정의
서비스 제공자 ID	RivetzNet에 의해 서비스 제공자에 할당되는 고유 식별자.
공개 키 암호화	
데이터 블랍	임의의 길이의 바이트들의 지정되지 않은 컬렉션으로서의 데이터.

[0347]

[0348] 복귀: 데이터 블랍 -- 임의의 길이의 바이트들의 지정되지 않은 컬렉션으로서의 데이터.

[0349] v. 복호화

논의	정의
서비스 제공자 ID	RivetzNet에 의해 서비스 제공자에 할당되는 고유 식별자.
데이터 블랍	임의의 길이의 바이트들의 지정되지 않은 컬렉션으로서의 데이터.

[0350]

[0351] 복귀: 데이터 블랍 -- 임의의 길이의 바이트들의 지정되지 않은 컬렉션으로서의 데이터.

[0352] b. 안드로이드 구현

[0353] 호스트 어댑터는 안드로이드에 대한 Rivetz 클라이언트의 표준 Java 부분이다. 호스트 어댑터는 호스트 어댑터가 크로스 앱 통신에 대한 표준 메커니즘인, 인텐트들을 통한 인터페이스라는 것을 드러낸다. 예를 들어:

```
public void connectRivet(String serviceProviderID, ByteArray instruction) {

    Intent intent = new Intent(com.rivetz.RivetActionExecute)

            .putExtra(com.rivet.RivetAction.EXTRA_SPID, serviceProviderID)

            .putExtra(com.rivet.RivetAction.EXTRA_INSTRUCTION, instruction);

    if (intent.resolveActivity(getPackageManager()) != null) {

        startActivity(intent);

    }

}
```

[0354]

[0355] 각각의 작동은 com.rivetz.RivetAction으로부터 이어받는 별도의 클래스로서 정의된다. 예를 들어:

```
public class RivetActionInstruction extends RivetAction { // RivetAction
extends Activity

        @Override

        protected void onCreate(Bundle savedInstanceState) {

                super.onCreate(savedInstanceState);



                // Get the intent that started this activity

                Intent intent = getIntent();

                int SpID =
intent.getStringExtra(com.rivet.RivetAction.EXTRA_SPID,0);

                ByteArray instruction =
intent.getStringExtra(com.rivet.RivetAction.EXTRA_INSTRUCTION,0);

                // call the corresponding JNI function

                result = Trustlet.RivetzActionPair(SpID,instruction);

        }
```

[0356]

[0357] TEE 어댑터는 디바이스 Rivet으로 명령어를 통과시키는 JNI(Java 네이티브 인터페이스) 코드를 정의한다.

[0358] i. 안드로이드 인텐트 문서화

[0359] 이러한 정의들은 공개 디스플레이를 위해 SDK 페이지들로 들어가게 된다. Rivetz 안드로이드 클라이언트 참조.

[0360] 새로운 안드로이드 인텐트의 제목: 

안드로이드 인텐트	정의
명령어_키 생성	지정된 타입의 키를 생성한다. Rivetz는 서비스 제공자에 고유한 국부 하드웨어 암호화된 저장 공간에 키를 저장한다. 키들은 장래 참조를 위해 명명된다.
명령어_복호화	명명된 키로 주어진 데이터 객체를 복호화한다.
명령어_키 삭제	서비스 제공자의 키 세트들로부터 키 명칭에 의해 식별되는 키를 제거한다.
명령어_암호화	명명된 키로 주어진 데이터 객체를 암호화한다. 일반적으로, 명명된 키는 명령어_로딩 키를 통해 로딩되는 공개 키와 함께 사용된다.
명령어_실행	디바이스에 서버 서명된 명령어를 제공한다. Rivet이 국부적인 서명되지 않은 요청들로 태스킹될 수 있지만, 이상적으로, 명령어들은 서비스 제공자 등록 동안 확립되는 서비스 제공자 키에 의해 서명된다.
명령어_키를 얻음	Rivet에 저장되는 명명된 키로부터 키 데이터를 얻는다. 결과들은 키 타입에 기반하여 달라질 것이다. 디바이스 하드웨어에 의해 보호되는 고유 키로 암호화되는 대칭 키들 및 개인 키들이 복귀된다.
명령어_포인터를 얻음	Rivetz.Net에 웹 요청들을 행하기 위해 사용될 수 있는 디바이스에 대한 임시 고유 포인터를 얻는다.
명령어_PUBPRV를 얻음	요약
명령어_PUBSIG를 얻음	요약
명령어_키ENUM	요약
명령어_로딩 키	명령어_암호화와의 사용을 위해 서비스 제공자 키 세트로 임의적인 공개 키를 로딩한다.
명령어_제공자 등록	서비스 제공자는 Rivet가 임의의 명령어에 응답하기 전에, 디바이스에 등록하거나 이것과 페어링할 필요가 있다. 이러한 프로세스는 Rivetz.net에 의해 중개되는 본질적으로 키 교환 세레모니이다.

[0361]

명령어_서명	명명된 키로 데이터의 블랍에 서명한다. 사용될 알고리즘이 키가 생성될 때, 확립된다.
명령어_서명TXN	명명된 코인(월렛) 키로 코인 트랜잭션에 서명한다.
명령어_검증	주어진 객체에 대한 시그니처를 검증한다. 결과 코드: Rivet.결과_정상은 시그니처가 패스되었다는 것을 나타낸다.

[0362]

[0363] c. Windows 구현

[0364] TBD

[0365] d. 사용 사례 참조

[0366] 국부 사용자 생성 - 어떤 서비스 제공자 인증도 행위자들인, 제품 행위자들로부터의 선택/생성 행위자들...이 주어지지 않는 경우들에서 Rivet의 사용을 인증할 수 있는 국부 엔티티를 확립한다.

[0367] · 어떤 것을 암호화함 - Rivetz는 텍스트 또는 이미지들을 암호화하는 메커닉들을 제공하지만 파트너들의 서비스에 대한 인터페이스가 메시징 애플리케이션이든..., 파트너들이 파트너들의 서비스에 대한 인터페이스를 기획하는 것을 기대한다.

[0368] ·

[0369] 21. 소켓 어댑터

[0370] 클라이언트 환경을 RivetzNet에 연결한다.

[0371] · 소켓 어댑터

[0372] · 구성 요소 콘텍스트

[0373] · 엔티티 책임

[0374] · 인터페이스 사양

[0375] · 연결

[0376] · 연결 해제

[0377] · 포인터를 얻음

[0378] · 명령

[0379] · 사용 사례 참조

[0380] a. 구성 요소 콘텍스트

[0381] b. 엔티티 책임

[0382] 새로운 엔티티의 제목: 

엔티티	정의
RivetzNet URL	본 발명이 RivetzNet를 호스팅하는 URL.
세션 객체	2개의 보안적 엔드포인트 사이의 일시적 세션에 대해 키들 및 다른 데이터를 정의한다.

[0383]

[0384] c. 인터페이스 사양

[0385] i. 연결

[0386] 서버와 연결을 개방한다. 서버는 이러한 세션에 할당되는 디바이스 포인터를 복귀시킬 것이다. 연결은 Rivet 어댑터가 시작할 때, 호출된다.

[0387] 논의들: 없음

[0388] 복귀들: 없음

[0389] ii. 연결 해제

[0390] 서버로부터 연결 해제하고 디바이스 포인터를 폐기한다.

[0391] 논의들: 없음

[0392] 복귀들: 없음

[0393] iii. 포인터를 얻음

[0394] 현재의 디바이스 포인터 또는 어떤 세션도 없으면, 널을 복귀시킨다.

[0395] 논의들: 없음

[0396] 복귀: 디바이스 포인터 -- 임의의 국부 애플리케이션에 의해 요청될 수 있는 디바이스에 대한 단명하는 포인터. 디바이스 포인터는 RivetzNet에 대한 현재의 소켓 세션을 식별할 수 있고 그러므로 디바이스 통신 채널을 확립하고 영구적 식별자인, 디바이스 ID를 검색하는데 사용될 수 있다.

[0397] iv. 명령

[0398] RivetzNet으로부터 명령어 기록을 수신하고, 명령어 기록을 rivet으로 패스하고, 응답 기록을 비동기로 발송한다. 모든 명령어는 명령어를 응답과 부합시키기 위해 RivetzNet에 의해 사용되는 고유 디스패치 ID가 붙어 있을 것이다. 일부 명령어가 TUI를 통한 사용자 상호 작용을 수반할 수 있고, 그러므로, 응답이 발송되기 전에, 상당한 경과된 시간을 초래할 수 있다는 것을 주목해야 한다.

논의	정의
디스패치 ID	RivetzNet으로부터 Rivet 어댑터에 의해 복귀되는 응답 기록으로 송신되는 명령어 기록에 부합시키는데 사용되는 고유 식별자.
서비스 제공자 ID	RivetzNet에 의해 서비스 제공자에 할당되는 고유 식별자.
명령어 기록	Rivetz 명령어는 식별된 디바이스 Rivet에 의해 처리되도록 타겟화되는 데이터 패키지이다. Rivetz 명령어는 Rivetz TEE 애플릿에서 일부 작동을 수행하도록 디바이스를 명령하기 위해 커맨드, 페이로드 및 필요한 시그니처들을 포함한다.
보고들	정의
디스패치 ID	RivetzNet으로부터 Rivet 어댑터에 의해 복귀되는 응답 기록으로 송신되는 명령어 기록에 부합시키는데 사용되는 고유 식별자.
응답 기록	복귀 상태 및 명령어 기록의 처리에 기인하는 페이로드.

[0399]

[0400] d. 사용 사례 참조

[0401] 22. TEE 어댑터

[0402] 이러한 구성 요소는 Trustonic 또는 Intel ME에서 실행되는 본 발명의 trustlet으로 커맨드들을 송신하는 전매 글루이다.

[0403] a. 설계 개념들

[0404] Trustonic 및 Intel ME 환경들은 동일한 기본 아키텍처를 따르며: 호스트 시스템은 데이터를 메모리 버퍼로 일련화하고 그 다음 처리할 TEE를 트리거한다. 이는 차단 (동기) 요청이다. 제어는 아마도 응답 데이터를 메모리 버퍼에 기록한 후에, TEE가 존재할 때, 복귀된다.

[0405] 본 발명의 TEE 코드가 하나보다 더 많은 것을 행할 수 있음에 따라, 데이터 구조체의 일부는 실행시킬 절차를 식별하기 위한 필요로 패스되었다. 이는 결국 데이터 구조체의 나머지가 해석되는 방법을 결정한다.

[0406] 마찬가지로, 실행되는 명령어는 함께 작동할 키들을 제공하는 콘텍스트 데이터를 필요로 한다. TEE가 어떤 네이티브 영속적 메모리도 가지지 않음에 따라, 데이터 기록들은 TEE에 의해 암호화되고 필요로 될 때, 저장하고 복귀시킬 TEE 어댑터에 주어진다. 기록들은 서비스 제공자에 대하여 저장되고 주어진 서비스 제공자에 고유한 디바이스 신원, 월렛 및 암호화 키들을 포함한다.

[0407] b. 구성 요소 도해

[0408] 모든 작업은 파라미터들 및 저장소로부터의 데이터가 공유된 메모리를 통하여 TEE 환경으로 패스될 구조체로 일련화되는 TEE 로더에서 일어난다.

[0409] i. TEE 통신 기록

[0410] 모든 요청의 경우, TEE 어댑터는 입력을 취하고, TEE에 대한 데이터 구조체를 패키징하고, 신뢰된 애플릿 환경 상의 실행을 호출한다. 실행이 완료될 때, 공유된 메모리는 응답 기록으로서 재구성된다. 임의의 복귀 데이터는 본래 호출 기능이 마련되고 서비스 제공자 기록은 다시 디스크로 저장된다.

[0411] c. 엔티티 책임

[0412] 새로운 엔티티의 제목: Submit

엔티티	정의
서비스 제공자 기록	TEE가 명령어를 처리할 때, TEE에 제공되는 서비스 제공자 콘텍스트 정보.

[0413]

[0414] d. 인터페이스 사양

[0415] i. 명령어를 처리함

[0416] 소켓 어댑터가 Rivetz 인코더로부터 명령어를 수신할 때, 소켓 어댑터에 의해 호출된다. 명령어는 분석하지 않고 TEE에 의해 직접 처리된 것으로 의미되는 패키징된 블랍이다.

논의	정의
서비스 제공자 ID	RivetzNet에 의해 서비스 제공자에 할당되는 고유 식별자.
명령어 기록	Rivetz 명령어는 식별된 디바이스 Rivet에 의해 처리되도록 타겟화되는 데이터 패키지이다. Rivetz 명령어는 Rivetz TEE 애플릿에서 일부 작동을 수행하도록 디바이스를 명령하기 위해 커맨드, 페이로드 및 필요한 시그니처들을 포함한다.

[0417]

[0418] TEE 어댑터는 서비스 제공자 기록을 로딩하고, 명령어 기록에 따라 서비스 제공자 기록을 메모리 버퍼로 일련화하고, 처리할 TEE를 트리거할 것이다. TEE 퇴거 시에, 서비스 제공자 기록은 다시 디스크로 기록되고 응답 블랍은 소켓 어댑터로 복귀된다.

[0419] ii. 암호화

[0420] 명명된 키를 사용하여 암호화하라는 국부 요청. 암호화 키들은 서비스 제공자 기록에 속하고 키 생성 명령어를 사용하여 생성된다.

논의	정의
서비스 제공자 ID	RivetzNet에 의해 서비스 제공자에 할당되는 고유 식별자.
키 명칭	Rivet에서 생성되는 키에 할당되는 임의적인 스트링.
데이터 블랍	임의의 길이의 바이트들의 지정되지 않은 컬렉션으로서의 데이터.

[0421]

[0422] iii. 복호화

[0423] 명명된 키를 사용하여 복호화하라는 국부 요청.

논의	정의
서비스 제공자 ID	RivetzNet에 의해 서비스 제공자에 할당되는 고유 식별자.
키 명칭	Rivet에서 생성되는 키에 할당되는 임의적인 스트링.
데이터 블랍	임의의 길이의 바이트들의 지정되지 않은 컬렉션으로서의 데이터.

[0424]

[0425] e. 안드로이드 구현

[0426] 안드로이드 구현은 안드로이드 NDK에 의해 구현되는 Java 네이티브 인터페이스(JNI)를 사용한다.

[0427] 디바이스 Rivet인, Trustonic 애플릿과 통신하기 위해, 본 발명은 안드로이드 JNI 코드를 사용할 필요가 있다. Rivet 작동 시에 발해지는 각각의 인텐트는 본 발명을 C++ 구현 환경으로 가져오는 정의된 상응하는 JNI 기능을 가질 것이다.

```
EXTERN_C JNIEXPORT jstring JNICALL

Java_com_rivetz_Trustlet_RivetzActionPair(JNIEnv *env, jobject obj, jstring
messageIn) {

      /* implementation */

}
```

[0428]

[0429] f. 사용 사례 참조

[0430] 23. 서비스 제공자 기록

[0431] TEE가 명령어를 처리할 때, TEE에 제공되는 서비스 제공자 콘텍스트 정보.

[0432] a. 구조체

[0433] 이러한 논제는 단지 개념들을 적어두기 위한 것이다.

속성	정의
서비스 제공자 ID	RivetzNet에 의해 서비스 제공자에 할당되는 고유 식별자.
키 기록	Rivet 어댑터 환경에 TEE 키들을 저장하는 영속적 객체. 각각의 키는 서비스 제공자를 대신하여 생성되고 이름 및 사용 규칙들이 주어진다.

[0434]

[0435] b. 실현

[0436] 이는 TEE 메모리 버퍼로 그리고 다시 이것 외로 용이하게 일련화될 수 있는 2진 데이터의 플랫 파일인 것으로 예상된다.

[0437] 세부 사항들 및 데이터 타입들이 GitHub에서의 소스 코드에서 정의되고 유지 관리된다. https://github.com/rivetz/RivetzEncoder/blob/master/riv_types.h 참조.

[0438] 24. Rivetz 프로토콜들

[0439] 디바이스 등록 프로토콜

[0440] 명령어 처리 프로토콜

[0441] Intercede 온보딩 프로세스

[0442] 25. 명령어 처리 프로토콜

[0443] a. 개요

[0444] 디바이스 Rivet의 대응물은 Rivetz 인코더이다. Rivetz 인코더는 서비스 제공자에 의해 서명되고/되거나 암호화되는 특정 디바이스에 의해 실행될 커맨드를 마련한다. 서비스 제공자 공개 키들은 RivetzNet에 의해 행해지는 페어링 프로세스 동안 디바이스로 미리 로딩된다. 이는 디바이스 Rivet이 요청의 근원을 입증하고, 필요하다면, 명령어의 내용들을 복호화하는 것을 가능하게 한다.

[0445] 명령어를 패키징하고 전달하는 시퀀스는 아주 간단하다. 서비스 제공자는 Rivetz 인코더 라이브러리들의 도움으로 명령어 기록을 생성한다. 명령어는 타입, 타겟 디바이스 및 페이로드를 포함한다. 명령어는 디바이스 키로 인코딩될 수 있고 서비스 제공자 키에 의해 서명되어야 한다. 디바이스 키는 디바이스 등록 기록을 검색함으로써 RivetzNet으로부터, 또는 블록 체인으로부터 직접 불러와진다.

[0446] 26. 디바이스 등록 프로토콜

[0447] a. 개요

[0448] 디바이스 등록은 본 발명의 전체 에코시스템이 의거하는 기초이다.

[0449] 27. Intercede 온보딩 프로세스

[0450] 이하의 것은 Rivetz가 디바이스 Rivet을 설치하기 위해 Intercede를 사용하는 것을 시작하도록 완료될 필요가 있을 단계들을 대략 설명한다.

[0451] 배경에 대한 Intercede 그룹 및 docs 참조.

[0452] · Intercede 온보딩 프로세스

[0453] · 키 설정:

[0454] · 디바이스 RIVET 애플리케이션을 구축함

[0455] · 실행

[0456] · 키 전송

[0457] · 개인화 마스터 키

[0458] · 키 검증

[0459] · 구매 영수증 키

[0460] a. 키 설정:

[0461] · 우선 테스트 키 전송(이를 TTK라 부를 것임)를 생성한다.

[0462] · 3개의 랜덤 256 비트값을 생성하고 3개의 랜덤 256 비트값을 셰어 1, 셰어 2, 셰어 3으로서 저장한다.

[0463] · TTK를 얻기 위해 셰어들 사이에서 XOR 작동(셰어 1 XOR 셰어 2 XOR 셰어 3)을 수행한다.

[0464] · 3개의 셰어 각각에 대한 파일들을 생성하고 Intercede가 Rivetz로 송신했던 3개의 PGP 키로 3개의 셰어를 개별적으로 암호화한다.

[0465] · 256 비트 테스트 개인화 마스터 키(TPMK)를 생성하고 256 비트 테스트 개인화 마스터 키(TPMK)를 어딘가의 Rivetz 코드에 저장한다.

[0466] · Intercede 문서에 설명하는 바와 같이 TTK로 TPMK를 암호화하고 TPMK를 이메일을 통하여 Intercede로 송신한다.

[0467] · 테스트 구매 영수증 키(TPRK)를 생성한다.

[0468] · Rosie 월렛 또는 본 발명이 원하는 모든 테스트 서비스 제공자에 대한 "고객 참조" 번호를 생성한다.

[0469] · TPRK의 공적 부분(이를 TPRPK라 부를 수 있음)을 Intercede로 송신한다.

[0470] b. 디바이스 RIVET 애플리케이션을 구축함

[0471] · 본 발명은 개인화 패키지를 수락할 수 있도록 현재의 디바이스 Rivet 소프트웨어를 변경할 것이다. 개인화 패키지는 TPMK로부터 끌어내어지는 키를 포함할 것이다.

[0472] · 각각의 개별 디바이스 Rivet에 대한 개인화 키를 끌어내는 Rivetz.net 서버측 상의 소프트웨어를 생성한다.

[0473] · 디바이스와 Rivetz.net 사이의 신뢰를 확립하기 위해 공유된 디바이스 Rivet 개인화 키를 사용하도록 Rivetz 권한 설정 프로토콜들을 업데이트한다. 이는 아마도 디바이스 Rivet이 새로운 디바이스 특정 키들을 생성하고 그러한 특정 디바이스 Rivet에 대한 개인화 키로 Rivetz.net에 대한 새로운 디바이스 특정 키들에 서명하는/이것을 암호화하는 것을 수반할 것이다.

[0474] · 디바이스 Rivet 및 개인화 패키지를 설치하는 것을 돕도록 본 발명의 실세계 애플리케이션(Rivet 어댑터)에서의 MyTAM 클라이언트 라이브러리를 포함한다.

[0475] c. 실행

[0476] i. 키 전송

[0477] 랜덤값들인, 셰어 1, 셰어 2, 셰어 2를 구축하기 위해:

[0478]
```
tr -cd [:alnum:] < /dev/urandom | head -c $(tr -cd 0-9 < /dev/urandom |
head -c 1) | sha256sum | tr -d ' -'
```

[0479] 이는 이하의 것처럼 보일 것이다:

[0480] a9f51566bd6705f7ea6ad54bb9deb449f795582d6529a0e22207b8981233ec58.

[0481] 이러한 커맨드가 행하는 것은 영숫자 문자들을 떼어내고, 결과를 (헤드를 갖는) 랜덤수의 문자로 길이를 줄이고, 그 다음 랜덤수의 문자를 sha256sum으로 송신하는 텍스트 처리 툴(tr)을 통해 리눅스 커널 랜덤 데이터를 송신하는 것이다. 마지막으로, 이는 후행 공백 및 하이픈을 제거하기 위해 다시 tr을 사용한다.

[0482] 이를 3 번 행하고 python 커맨드 라인 호출을 사용하여 함께 결과들을 XOR를 행한다:

[0483]
```
python -c 'print "{:x}".format(

int("bb65b75d83d82065b17929affd23e8f26f9e134ff90646e1fd087eb4339b89fe",16)
^

int("e5568e87e6fd44b373fa92c361f5c5c37ce5f4ddf97cefe1177b3d3720912854",16)
^

int("a9f51566bd6705f7ea6ad54bb9deb449f795582d6529a0e22207b8981233ec58",16))
'
```

[0484] 이는 이하를 야기한다:

[0485] f7c62cbcd842612128e96e2725089978e4eebfbf655309e2c874fb1b01394df2

[0486] 이것이 하는 것은 16진 스트링들 각각을 int로 계산하고, 그것들을 함께 XOR을 행하고, 그 다음 결과를 다시 16진으로 형식화하는 것이다.

[0487] 이러한 파일들이 모두 ASCII 16진 표현이라는 것을 주목해야 한다. 2진수로 변환하기 위해 이하를 행한다:

[0488]
```
cat share1 | xxd -r -p > share1.bin
```

[0489] 이를 다 함께 놓으면,

```
tr -cd [:alnum:] < /dev/urandom | head -c $(tr -cd 0-9 < /dev/urandom |
head -c 1) | sha256sum | tr -d ' -' > share1

tr -cd [:alnum:] < /dev/urandom | head -c $(tr -cd 0-9 < /dev/urandom |
head -c 1) | sha256sum | tr -d ' -' > share2

tr -cd [:alnum:] < /dev/urandom | head -c $(tr -cd 0-9 < /dev/urandom |
head -c 1) | sha256sum | tr -d ' -' > share3

python -c 'print "{:x}".format(int(open("share1","r").read(),16) ^
int(open("share2","r").read(),16) ^ int(open("share3","r").read(),16))' >
TTK
```

[0490]

[0491] 그 다음 각각의 단편에 대해:

```
gpg --import recipient.asc

cat share1 | xxd -r -p > share1.bin

gpg -o encrypted_share_for_recipient.gpg --encrypt -r <KEY-ID> share1.bin
```

[0492]

[0493] ii. 개인화 마스터 키

[0494] 1. 랜덤수를 생성함

[0495] 2. 2진수로 변환함

[0496] 3. 키 전송으로 암호화하고 그 다음 Intercede로의 전달을 위해 16진 형식으로 송신함

```
tr -cd [:alnum:] < /dev/urandom | head -c $(tr -cd 0-9 < /dev/urandom |
head -c 1) | sha256sum | tr -d ' -' > TPMK

cat TPMK | xxd -r -p > TPMK.bin

openssl enc -aes-256-ecb -in TPMK.bin -nopad -K `cat TTK` | xxd -p -c 256 >
TPMK.enc.hex
```

[0497]

[0498] iii. 키 검증

[0499] 체크값(KCV)은 계산되고 Intercede로 송신될 수도 있다. 선택적 체크값은 Intercede HSM으로 불러와지면, 개인화 마스터 키가 정확하다는 것을 보장하며 - 체크값은 이하와 같이 컴퓨팅된다.

[0500] · 2진수 제로들의 하나의 블록(16 바이트)을 암호화하기 위해 (암호화되지 않은) 개인화 마스터 키를 사용한다. (ECB 모드를 사용함, 추가 없음.)

[0501] · 출력의 첫 번째 3 바이트는 체크값(KCV)이다. KCV를 Intercede로 송신한다.

[0502] · Intercede에서 키를 MyTAM으로 불러오는 프로세스는 KCV(공급된다면)를 검증하고, 키 교환이 정확하게 수행되었다는 부가 검증을 제공할 것이다.

```
echo 00000000000000000000000000000000 | xxd -p -r | openssl enc -aes-256-
ecb -nopad -K `cat TPMK` | xxd -p -c 256 | cut -b -6 > TPMK.kcv
```

[0503]

[0504] iv. 구매 영수증 키

[0505] 이는 인 앱 구매들에 대한 Google Play 영수증 키를 모방하는 것으로 추측된다. 키는 권한 설정 동안 디바이스 SUID에 서명하는데 사용된다. Intercede는 이것을 "구매"의 영수증으로서 사용한다.

[0506]
```
openssl genrsa -out TPRK.pem 2048

openssl rsa -in TPRK.pem -pubout > TPRPK.pem
```

[0507] 어것은 파일 TPRK.pem에서 2048 비트 RSA 키를 생성하고 그 다음 Intercede로 송신될 TPRPK.pem으로 공개 키를 추출한다.

[0508] openssl.org에서: "PEM 형태는 디폴트 형식이며: PEM 형태는 부가 헤더 및 푸터 라인들로 인코딩되는 DER 형식 base64로 구성된다. 입력 시에, PKCS#8 형식 개인 키들이 또한 수락된다."

[0509] Google Play 문서에서: "Google Play에 의해 생성되는 Base64 인코딩된 RSA 공개 키는 2진수 인코딩 X.509 대상 공개 키 정보 DER 시퀀스 형식으로 있다. Google Play 라이센싱으로 사용되는 것은 동일한 공개 키이다."

[0510]
```
openssl genrsa -out TPRK.pem 2048

openssl rsa -in TPRK.pem -outform der -pubout > TPRPK.der
```

[0511] 이는 2진 형식 키를 전달한다.

[0512] 28. Rivetz 사용 사례들

[0513] Rivetz는 디바이스로 단순한 한층 더 중대한 트랜잭션들을 달성하기 위해 파트너들에게 SDK를 제공한다. 이는 메시지들에 대한 승인 내지 비트코인 서명에 걸친다. 인터페이스는 시스템 인터페이스이지만, 일부 서비스는 사용자에게 핀 입력, 시각 확인 등을 보증할 것이다.

[0514] a. 사용 사례들

[0515] 새로운 사용 사례의 제목: Submit

사용 사례	정의
비트코인 계정 생성	디바이스 하드웨어에서 새로운 월렛 계정 id를 생성한다.
키 생성	양 서명 및 암호화에 대한 디바이스 Rivet에서의 키 쌍을 생성한다.
국부 사용자 생성	어떤 서비스 제공자 인증도 주어지지 않는 경우들에서 Rivet의 사용을 인증할 수 있는 국부 엔티티를 확립한다.
어떤 것을 복호화함	암호화된 객체 및 키 명칭을 고려하여, TUI 디스플레이를 위해 또는 요청자로 복귀시키기 위해 객체를 복호화한다.
어떤 것을 암호화함	Rivetz는 텍스트 또는 이미지들을 암호화하는 메커닉들을 제공하지만 파트너들의 서비스에 대한 인터페이스가 메시징 애플리케이션이든 아니면 어떤 다른 것이든, 파트너들이 파트너들의 서비스에 대한 인터페이스를 기획하는 것을 기대한다.
Rivetz에의 디바이스 등록	Rivet이 임의의 것을 할 수 있기 전에, Rivet은 RivetzNet에 등록할 필요가 있다. 등록은 고유 신원 키의 생성을 야기한다.
서비스 제공자에의 디바이스 등록	서비스 제공자는 디바이스가 임의의 요청들에 응답하기 전에, 그러한 디바이스에 등록되는 서비스 제공자의 서비스 제공자 ID 및 공개 신원 키를 가질 필요가 있다.
Rivetz에의 서비스 제공자 등록	Rivetz 시스템으로 코딩할 것을 요구하는 누구나 서비스 제공자로서 등록할 필요가 있다.
보안적 확인 요청을 송신함	타겟 엔드포인트 디바이스로 전달되고 이용 가능하다면, 보안적 디스플레이로 사용자에게 표시될 짧은 메시지를 패키징한다. 통신되는 것은 확인이 유효하다는 것을 보장하기 위한 방식들 둘 다에 서명된다. 메시지는 이미지 또는 텍스트일 수 있다.
비트코인 트랜잭션 서명	(근원 계정이 타겟 디바이스 하드웨어에 의해 소유되는) 충분히 형성된 비트코인 트랜잭션을 고려하여, 트랜잭션에 서명하고 트랜잭션을 복귀시킨다. 대부분의 경우에, 이는 또한 이용 가능하다면, 보안적 디스플레이 또는 달리 적어도 통상의 디스플레이로의 확인을 위해 사용자에게 프롬프트하는 것을 수반할 것이다.

[0516]

어떤 것을 서명함	명명된 키 및 객체 참조를 고려하여, 객체의 서명된 해시를 복귀시킨다.
사용자가 잊어 버려진 디바이스 핀을 되찾음	요약
어떤 것을 검증함	명명되거나 주어진 키로 객체 상의 시그니처를 검증한다.

[0517]

[0518] b. 행위자들

[0519] 새로운 행위자의 제목: Submit

행위자	정의
계정 대표자	서비스 제공자와의 관계에 책임이 있는 Rivetz 고용인.
서비스 제공자	서비스 제공자들은 서비스 제공자들 자체의 서비스들을 강화시키기 위해 Rivetz에 제공되는 능력들을 사용한다. 서비스 제공자들은 본 발명의 파트너들 및 주수입원이다.
서비스 사용자	서비스 사용자는 본 발명의 서비스의 주특징/기능과 관계되는 어떤 자이다.
시스템 관리자	시스템 관리자는 본 발명의 서비스의 설치, 구성 및 유지 관리에 관계한다.
신뢰된 애플리케이션 관리자	신뢰된 실행 환경(TEE)으로 신뢰된 애플리케이션을 로딩하고 보증할 수 있는 엔티티.

[0520]

[0521] 29. 신뢰된 애플리케이션 관리자

[0522] 신뢰된 실행 환경(TEE)으로 신뢰된 애플리케이션을 로딩하고 보증할 수 있는 엔티티.

[0523] a. 정의

[0524] Trustonic의 세계에서, Giesecke 및 Devrient 및 Intercede 그룹은 TAM들로서 확립된다.

[0525] 30. 서비스 사용자

[0526] 서비스 사용자는 본 발명의 서비스의 주특징/기능과 관계되는 어떤 자이다.

[0527] a. 정의

[0528] 31. 시스템 관리자

[0529] 시스템 관리자는 본 발명의 서비스의 설치, 구성 및 유지 관리에 관계한다.

[0530] a. 정의

[0531] 32. 계정 대표자

[0532] 서비스 제공자와의 관계에 책임이 있는 Rivetz 고용인.

[0533] a. 정의

[0534] 33. 서비스 제공자

[0535] 서비스 제공자들은 서비스 제공자들 자체의 서비스들을 강화시키기 위해 Rivetz에 제공되는 능력들을 사용한다.

[0536] 정의

[0537] 서비스 제공자들은 본 발명으로 사업을 하거나, 보다 상세하게는, 본 발명의 API들에 액세스하고 리벳화된 디바이스들로 타겟화되는 명령어들에 서명하기 위해 RivetzNet에 등록될 필요가 있다.

[0538] a. 데모 서비스 제공자

[0539] 이른 테스트 및 시험을 위해 개발자들에게 용이하게 배포될 수 있는 서비스 제공자 ID를 가질 필요가 있다는 것은 분명하다. 본 발명은 이미, 그러나 MarkHoblit가 내장되었던 랜덤 UUID로 이것을 행하고 있다. 예를 들어:

```
Intent intent = new Intent(Rivet.RIVET_INTENT)

              .putExtra(Rivet.EXTRA_INSTRUCT, Rivet.INSTRUCT_CREATEKEY)

              .putExtra(Rivet.EXTRA_SPID, "98f88054-f98c-440c-81aa-
77fa70a31116-fbca7c00-0602-4c1f-a354-820ae9ec46b9")

              .putExtra(Rivet.EXTRA_KEYTYPE, Rivet.KEYTYPE_ECDSA_DEFAULT)

              .putExtra(Rivet.EXTRA_KEYNAME,"MyKey");
```

[0540]

[0541] 데모 SPID로 활성화되는 디바이스가 생산 Rivet처럼 Intercede 및 Trustonic에 대한 저작권 사용료를 발생시킬 것이라는 점이 주목되어야 한다.

[0542] 34. Rivetz에의 서비스 제공자 등록

[0543] Rivetz 시스템으로 코딩할 것을 요구하는 누구나 서비스 제공자로서 등록할 필요가 있다.

[0544] 초기 등록은 RivetzNet(http://rivetz.com/docs/registration.html) 상의 서식을 채우는 것만큼 단순하다.

[0545] _____

[0546] a. 행위자들

[0547] 서비스 제공자, 계정 대표자

[0548] b. 설명

[0549] 1. 서비스 제공자는 국부 공개/개인 키들을 생성한다.

[0550] 2. 서비스 제공자는 rivetz.com(http://rivetz.com/docs/registration) 상의 HTTP 서식으로 가고 이하의 정보를 입력한다:

[0551] · 기업 이름

[0552] · 연락: 이름, 성, 위치, 이메일, 전화기

[0553] · 기업 웹사이트

[0554] · 기업 주소: 거리, 도시, 주/도, 국가

[0555] 3. 서비스 제공자는 서비스 동의의 항들에 "나는 수락함"을 클릭한다.

[0556] 4. 서비스 제공자는 비밀 번호를 선택하고 비밀 번호를 확인한다(사용자 이름은 주어진 연락 이메일일 것이다).

[0557] · 이것이 이후에 디바이스 승인으로 대체될 수 있다는 것을 서비스 제공자에게 말한다.

[0558] 5. 서비스 제공자는 공개 키를 업로드하도록 요청된다.

[0559] · 이는 건너 뛰어지고 이후에 행해질 수 있다.

[0560] · 본 발명은 또한 이러한 업로드보다 공개 키를 얻는 더 보안적인 방식들을 제공할 것이다.

[0561] 6. 키가 제공되면, 그 다음 SPID(서비스 제공자 ID)가 생성되고 고객으로 이메일이 보내진다.

[0562] · 어떤 키도 제공되지 않으면, 이메일 확인이 미결정 메시지 및 키를 제공할 시의 명령어들과 함께 송신된다.

[0563] 7. 계정 대표자는 새로운 등록의 통지를 수신할 것이다.

[0564] · 이 시점에서, 데이터는 판매 부서로 로딩될 수 있고 계정 응답은 개인적으로 더 알아보는 것을 선택할 수 있다.

[0565] i. 변화: 새로운 서비스 제공자는 키를 제공하도록 복귀됨

[0566] 1. 서비스 제공자는 이메일 및 비밀 번호로 로그인한다.

[0567] 2. 서비스 제공자는 계정의 "미결정" 상태를 메모한다.

[0568] 3. 서비스 제공자는 미결정 상태를 고정시키는 것을 클릭하고 서비스 제공자의 공개 키에 대한 입력 박스로 프롬프트된다.

[0569] 4. 키가 발송되면, SPID가 생성되고 서비스 제공자 연락 이메일로 이메일이 보내진다.

[0570] 5. 계정은 더 이상 미결정이지 않다.

[0571] 6. 계정 대표자는 계정의 변경이 통지된다.

[0572] c. 비고

[0573] 35. 사용자가 잊어 버려진 디바이스 핀을 되찾음

[0574] 요약

[0575] _____

[0576] a. 행위자들

[0577] 제품 행위자들로부터의 선택/생성 행위자들

[0578] b. 설명

[0579] c. 비고

[0580] 36. 어떤 것을 검증함

[0581] 명명되거나 주어진 키로 객체 상의 시그니처를 검증한다.

[0582] 어떤 것을 암호화함 같이, 어떤 것을 검증함은 공개 키를 사용함에 따라, 보안적 프로세스가 아니다. 어떤 것을 검증함은 편의를 위해 제공된다. 어떤 것을 검증함의 대응물인, 어떤 것을 서명함 참조.

[0583] _____

[0584] a. 행위자들

[0585] 서비스 제공자

[0586] b. 설명

[0587] c. 비고

[0588] 웹 홈 > 제품 관점 > 제품 사용 사례들 > Rivetz 사용 사례들 > 키 생성

[0589] 37. 키 생성

[0590] 양 서명 및 암호화에 대한 디바이스 Rivet에서의 키 쌍을 생성한다.

[0591] _____

[0592] a. 행위자들

[0593] 서비스 제공자

[0594] b. 설명

[0595] Rivetz의 주목적은 엔드포인트 디바이스들 내에서 키들을 보안하고 적용하는 것이다. 암호화(프라이버시) 키들 또는 서명(신원) 키들은 TEE에서의 암호화 툴들을 사용하여 생성되고 TEE의 저장 키를 사용하여 디바이스 상에 보안적으로 저장된다. 비트코인 어드레스 키들이 마찬가지로 유지 관리되지만 미묘한 차이들을 갖는다(비트코인 계정 생성 참조).

[0596] 모든 키는 서비스 제공자의 맥락에서 생성된다. 즉, 모든 키는 요청된 모든 키의 생성을 요청하였던 서비스 제공자 ID와 함께 저장된다. 모든 키는 서비스 제공자 ID의 맥락으로 고유한 이름이 주어진다.

[0597] 키가 생성될 때, 키의 사용에 대한 규칙들은 임의의 조합으로 지정된다. 이들은:

[0598] · 키의 생성자(서비스 제공자)에 의해 키를 적용하도록 서명된 요청을 필요로 하며,

[0599] · 신뢰된 사용자 인터페이스를 통해 키를 적용하도록 사용자 확인을 필요로 하며,

[0600] · TUI에서 표시되는 결과를 필요로 한다.

[0601] TUI에서 표시되는 결과를 갖는 것이 무엇을 의미하는 지에서의 더 많은 논의에 대해 어떤 것을 복호화함 및 어떤 것을 검증함 참조.

[0602] c. 비고

[0603] 38. 비트코인 계정 생성

[0604] 디바이스 하드웨어에서 새로운 월렛 계정 id를 생성한다.

[0605] a. 행위자들

[0606] 서비스 제공자

[0607] b. 설명

[0608] 모든 리벳화된 키 같이, 새로운 비트코인 계정은 서비스 제공자의 콘텍스트 내에서 생성되고 이름이 주어진다. 서비스 제공자 앱은 이러한 이름을 숨기거나 이러한 이름을 최종 사용자에게 특징으로서 제공할 수 있다.

[0609] 비트코인 어드레스를 생성할 때, 서비스 제공자는 계정이 트랜잭션에 서명하도록 TUI 확인을 필요로 하는지 여부를 지정해야 한다.

[0610] c. 비고

[0611] 39. 어떤 것을 암호화함

[0612] Rivetz는 텍스트 또는 이미지들을 암호화하는 메커닉들을 제공하지만 파트너들의 서비스에 대한 인터페이스가 메시징 애플리케이션이든 아니면 어떤 다른 것이든, 파트너들이 파트너들의 서비스에 대한 인터페이스를 기획하는 것을 기대한다.

[0613] 복호화 키들은 복호화된 객체의 TUI 디스플레이를 필요로 하도록 표시될 수 있다.

[0614] MJS> 이는 TUI 확인을 필요로 하는 것과 별개이라는 것을 주목해야 한다.

[0615] ———————————————————————

[0616] a. 행위자들

[0617] 서비스 사용자, 서비스 제공자

[0618] b. 설명

[0619] Rivet 어댑터는 타겟 디바이스의 공개 키를 가져야 할 것이다. 이는 디바이스들의 페어링 동안 서비스 제공자에 의해 직접 제공되거나 디바이스 Rivet에서 앞서 기록된다. 암호화측 상에서, 디바이스 Rivet은 작동이 공개 키 작동만이므로, 수반될 필요가 없다. 그럼에도 불구하고 암호화측 상에서, 호스트 어댑터 인터페이스 (또는 Rivetz 인코더)에서의 기능에 대한 입력들은 이하를 포함한다:

[0620] * 타겟 디바이스 ID 또는 타겟 디바이스 정적 공개 암호화 키(암호화 키는 암호화를 수행하는 엔티티에 의해 알려져야 함). * (선택적) 암호화될 데이터.

[0621] 가장 단순한 예시화에서, Rivetz는 ECDH 작동만을 제공한다. 이것에 행해질 때, 암호화되거나 복호화될 데이터는 Rivetz 소프트웨어로 패스되지 않고, 대신에 Rivetz 소프트웨어는 ECDS 작동으로부터 공유된 기밀을 단순히 출력할 것이다. 그 다음 그러한 공유된 기밀을 사용하여 데이터 암호화를 수행하는 것은 외부 소프트웨어에 달려 있다.

[0622] c. 비고

[0623] 40. 보안적 확인 요청을 송신함

[0624] 타겟 엔드포인트 디바이스로 전달되고 이용 가능하다면, 보안적 디스플레이로 사용자에게 표시될 짧은 메시지를 패키징한다. 통신되는 것은 확인이 유효하다는 것을 보장하기 위한 방식들 둘 다에 서명된다. 메시지는 이미지

또는 텍스트일 수 있다.

[0625] _____

[0626] a. 행위자들

[0627] 서비스 제공자, 서비스 사용자

[0628] b. 설명

[0629] 보안적 확인 요청의 가치는 메시지가 일부 다른 디바이스에 의해 확인될 수 있는 의도되는 기회 이외의 매우 적은 기회(존재한다면)가 있다는 것을 인지하는 것이다. 게다가, 디바이스가 소스가 나타내어지게 될 수만 있는 확인을 표시하고 있다는 것이다. 이를 달성하는 것은 메시지가 네트워크의 와일드 프린지(wild fringe)(사용자의 디바이스들)에서의 디스플레이를 위해 처리되고 제공되고 있을 때, 별다른 어떤 것도 일어나고 있지 않는다는 것을 보장하기 위해 디바이스 및 서비스 제공자 둘 다 그리고 디바이스에서의 TEE로부터의 키들의 등록을 필요로 한다.

[0630] 서비스 제공자는 메시지 및 타겟 디바이스를 단순히 표명하는 것을 기대하고 응답을 대기할 것이다. 키잉 인프라 구조체는 소스 코드가 신뢰되는 한은 수학만이 행해지고 있다는 것을 보장하기 위해 모든 당사자 및 대중에 대해 독립적이어야 한다.

[0631] c. 비고

[0632] 41. 어떤 것을 서명함

[0633] 명명된 키 및 객체 참조를 고려하여, 객체의 서명된 해시를 복귀시킨다.

[0634] _____

[0635] a. 행위자들

[0636] 서비스 제공자

[0637] b. 설명

[0638] 신원 키들이 키 생성에 설명하는 바와 같은 키 사용 규칙들을 따를 것이라는 것을 주목해야 한다.

[0639] c. 비고

[0640] 42. Rivetz에의 디바이스 등록

[0641] Rivet이 임의의 것을 할 수 있기 전에, Rivet은 RivetzNet에 등록할 필요가 있다. 등록은 고유 신원 키의 생성을 야기한다.

[0642] 등록은 디바이스 Rivet이 보안적 환경에서 적절하게 실행되고 있다는 것을 보장하는 신뢰된 애플리케이션 관리자로부터의 보증에 의존한다. (이상적으로, 신뢰된 애플리케이션 관리자에 의해 확립되는 키는 디바이스 등록 키에 국부적으로 서명할 것이다.)

[0643] a. 행위자들

[0644] 신뢰된 애플리케이션 관리자

[0645] b. 설명

[0646] 디바이스 등록 프로토콜 참조.

[0647] 등록은 Rivet 어댑터가 작동되는 처음에 일어나고 키 쌍이 Rivet에서 생성되는 것 및 공개 키가 RivetzNet와 공유되는 것을 야기한다. 디바이스가 등록되면, 디바이스는 디바이스가 제대로 기능을 하는 언제든지 RabbitMQ 소켓을 통해 RivetzNet로 연결되는 것을 시도할 것이다.

[0648] 1. 디바이스는 국부 공개/개인 키들을 생성한다.

[0649] 이러한 키들은 서비스 제공자 "Rivetz"에의 신원 키로서 국부적으로 저장될 것이다.

[0650] 2. 디바이스는 rivetz.net에 HTTP REST 호출을 행하여 고유 식별자로서 공개 키의 시그니처로 등록을 요청한다.

[0651] RivetzNet은 신뢰된 애플리케이션 관리자(TBD)에 의해 제공되는 프로토콜을 통해 요청의 유효성을 테스트할 필요가 있다.

[0652] 3. 디바이스는 디바이스의 고유 디바이스 ID로 디바이스가 이제 등록된다는 것을 나타내는 (또는 디바이스가 앞서 등록되었다는 것을 나타내는) 응답 및 유입되는 커맨드들을 청취할 RabbitMQ 큐 명칭을 수신한다.

[0653] 4. 디바이스는 지정된 큐 상에서 유입되는 커맨드들을 청취하기 위해 RabbitMQ를 시동한다.

[0654] c. 비고

[0655] 43. 비트코인 트랜잭션 서명

[0656] (근원 계정이 타겟 디바이스 하드웨어에 의해 소유되는) 충분히 형성된 비트코인 트랜잭션을 고려하여, 트랜잭션에 서명하고 트랜잭션을 복귀시킨다. 대부분의 경우에, 이는 또한 이용 가능하다면, 보안적 디스플레이 또는 달리 적어도 통상의 디스플레이로의 확인을 위해 사용자에게 프롬프트하는 것을 수반할 것이다.

[0657] _____

[0658] a. 행위자들

[0659] 서비스 제공자, 서비스 사용자

[0660] b. 설명

[0661] c. 비고

[0662] 44. 국부 사용자 생성

[0663] 어떤 서비스 제공자 인증도 주어지지 않는 경우들에서 Rivet의 사용을 인증할 수 있는 국부 엔티티를 확립한다.

[0664] _____

[0665] a. 행위자들

[0666] 제품 행위자들로부터의 선택/생성 행위자들

[0667] * 디바이스 Rivet

[0668] * TEE 어댑터

[0669] * Rivetz.net(선택적)

[0670] b. 설명

[0671] 디바이스 Rivet의 빠르고 용이한 사용을 가능하게 하기 위해, 디바이스 Rivet은 "국부 사용자"의 생성을 가능하게 할 수 있다. 국부 사용자는 인증된 서비스 제공자가 아니고, 어느 정도로 디바이스 Rivet에 액세스하는 것이 가능해지는 엔티티인 것으로 정의된다. 서비스 제공자가 비트코인 키들을 생성하고 관리하고 다른 서비스들을 제공하는 것이 가능해질 수 있지만, 국부 사용자는 일정 작동들만을 수행하도록 인증될 수 있다. 이러한 작동들은 이하를 포함할 수 있다:

[0672] * 암호화 키들을 생성하고 사용하는 것

[0673] * 시그니처 키들을 생성하고 사용하는 것

[0674] 국부 사용자의 특성들은 이하와 같다:

[0675] - 국부 사용자에 대한 인증은 국부 플랫폼 상에서 초기에 유지될 것이지만, 어떤 다른 곳에서 이후에 보호될 수 있다.

[0676] - 국부 사용자는 Rivetz.net에 의해 선택적으로 인증된다.

[0677] - 국부 사용자는 실제 사람 사용자 또는 애플리케이션으로부터 숨겨질 수 있다. 국부 사용자는 Rivet 어댑터 내에서 관리될 수 있다.

[0678] - 국부 사용자에 대한 인증의 보호는 사용자 비밀 번호로의 암호화 또는 일부 다른 보호 메커니즘의 사용을 포함하도록 시간이 지남에 따라 강화될 수 있다.

[0679] - 애플리케이션 관점에서, 호스트 어댑터는 국부 사용자와 연관된 키들이 호스트 어댑터를 통해서 이외의 임의의 인터페이스를 통해 액세스 가능하지 않다는 사실과 다른, 투명한 국부 사용자의 개념을 만드는 인터페이스를 제공한다.

[0680] "국부 사용자"가 반드시 외부 관점에서는 아닌, 디바이스 Rivet 관점에서의 사용자이므로, "국부 사용자"의 이름을 고려하는데 세심해야 한다. 한가지 개념은 국부 사용자가 TEE 어댑터에 의해 처리되다는 것이다. TEE 어댑터는 디바이스 Rivet으로 공유된 기밀을 확립하거나 디바이스 Rivet으로 국부 사용자를 인증하는 공개 키를 생성한다.

[0681] c. 비고

[0682] 45. 국부 사용자

[0683] 국부 사용자는 형식적 서비스 제공자로부터의 참여 없이 디바이스 Rivet에 액세스할 수 있는 엔티티이다. 즉, 국부 사용자는 전형적 서비스 제공자와 상이한 역할이고 하나의 특정 디바이스 Rivet에만 액세스할 수 있는 각각의 디바이스 Rivet에 대한 상이한 국부 사용자가 있을 수 있다는 것이 예상될 수 있다.

[0684] 국부 사용자의 권한 설정에 대한 일부 결정이 행해질 것이지만, 하나의 가능성은 Rivetz.net이 (예를 들어, "페어링" 작동을 통해) 전형적 서비스 제공자로 행해질 수 있는 동일한 방식으로의 권한 설정 단계 동안 국부 사용자를 인증한다는 것이다. 이러한 경우라면, Rivetz는 디바이스 Rivet 서비스들에 액세스할 수 있는 자를 통한 제어를 여전히 유지 관리하고, 또한 장래에, (일부 신뢰된 엔티티에 의해 강하게 보호되고 제어되는 국부 사용자에 대한 인증을 보장함으로써) 국부 사용자 역할에 대한 액세스를 통해 일부 강한 보호를 제공할 수 있다.

[0685] 국부 사용자가 인증되는 방식에 대한 결정이 또한 행해질 것이다. 단순함을 위해, 본 발명은 국부 사용자에 의한 작동들이 (예를 들어, 시그니처 작동을 통해) 서비스 제공자로부터의 작동들과 동일한 종류의 인증을 필요로 한다는 것을 필요로 할 수 있거나, 단기적으로는, 본 발명은 단순히 국부 사용자가 공유 기밀(예를 들어, 비밀번호, 패스프레이즈 또는 랜덤값)을 활용하는 것을 가능하게 할 수 있다.

[0686] · 국부 사용자

[0687] 46. 서비스 제공자에의 디바이스 등록

[0688] 서비스 제공자는 디바이스가 임의의 요청들에 응답하기 전에, 그러한 디바이스에 등록되는 서비스 제공자의 서비스 제공자 ID 및 공개 신원 키를 가질 필요가 있다.

[0689] 명명된 키(신원, 프라이버시 또는 코인)가 서명된 요청을 필요로 하지 않는 경우들에서도, 요청 당사자의 ID는 디바이스에 알려져야 한다. RivetzNet은 디바이스와 서비스 제공자 사이의 관계를 보증하는데 책임이 있다. 이러한 방식으로, 본 발명은 에코시스템을 통한 일부 제어를 유지 관리한다. 이는 또한 서비스 제공자 키들의 사용, 백업 및 이동에 관하여 최종 사용자들에게 본 발명이 서비스들을 제공하는 것을 가능하게 한다.

[0690] ———————————————————————————

[0691] a. 행위자들

[0692] 서비스 제공자

[0693] b. 설명

[0694] 1. 국부 서비스 제공자 앱은 디바이스 포인터에 대해 Rivet 어댑터에 요청을 한다.

[0695] 2. 디바이스는 공개 키뿐만 아니라 새로운 디바이스 포인터 및 디바이스 ID(비고: 여기서 승인을 필요로 하며....앞서와 마찬가지로 공개 키 또는 API 키를 사용할 수 있음)로 RivetzNet에 HTTP REST 호출을 한다.

[0696] 3. 서버로부터의 응답은 유입되는 서비스 제공자의 공개 키를 대기할 RabbitMQ 큐를 포함한다.

[0697] 4. 서비스 제공자는 디바이스 포인터를 서비스 제공자의 서버들로 패스한다.

[0698] 5. 서비스 제공자는 디바이스 포인터 및 SP의 공개 키로 HTTP REST 호출을 한다.

[0699] 6. 서비스 제공자에 대한 응답은 디바이스 공개 키를 포함한다.

[0700] 7. 서비스 제공자의 공개 키는 디바이스로 푸싱된다.

[0701] c. 비고

[0702] 47. 어떤 것을 복호화함

[0703] 암호화된 객체 및 키 명칭을 고려하여, TUI 디스플레이를 위해 또는 요청자로 복귀시키기 위해 객체를 복호화한다.

[0704] ---

[0705] a. 행위자들

[0706] 서비스 제공자

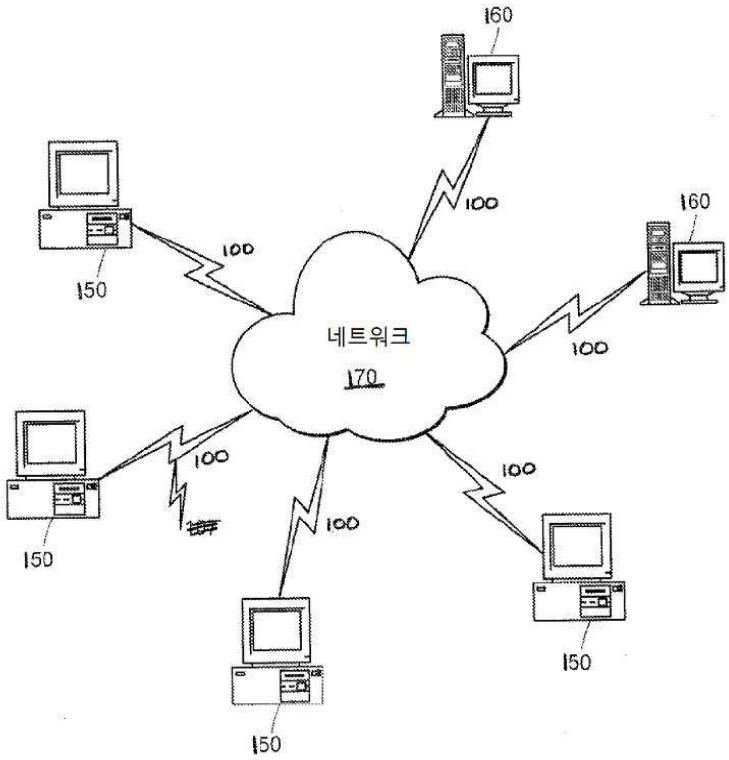
[0707] b. 설명

[0708] 프라이버시 키 쌍이 생성될 때, 프라이버시 키 쌍은 요청이 TUI를 통해 사용자에 의해 서명되고/되거나 확인될 필요가 있는지 여부를 지정하는 키 사용 규칙들로 표시될 필요가 있다. 게다가, 키는 TUI 디스플레이에 대해서로서만 지정될 수 있어 키가 복호화하는 어떤 것도 보안적 세계에 머무른다는 것을 의미한다.

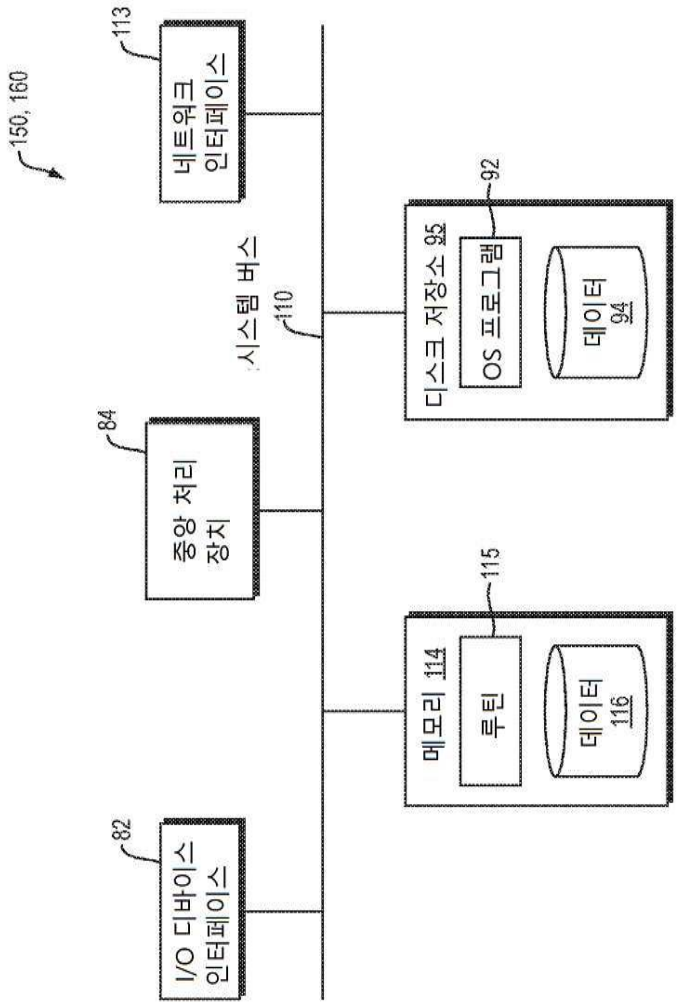
[0709] c. 비고

도면

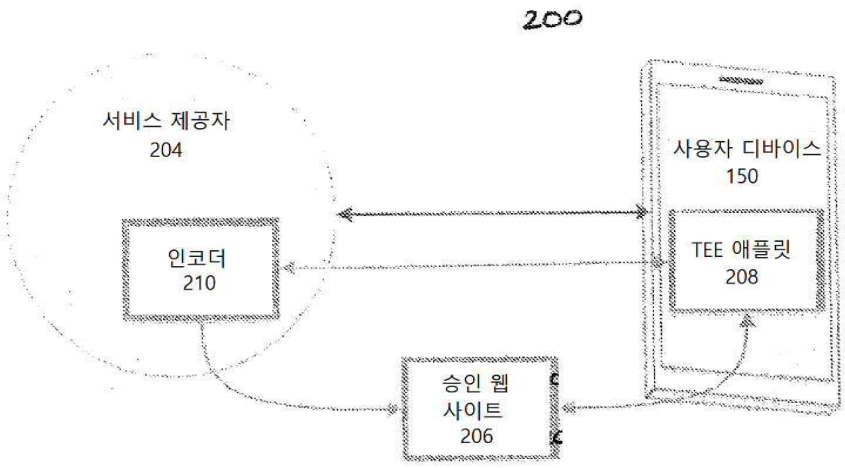
도면1a



도면1b

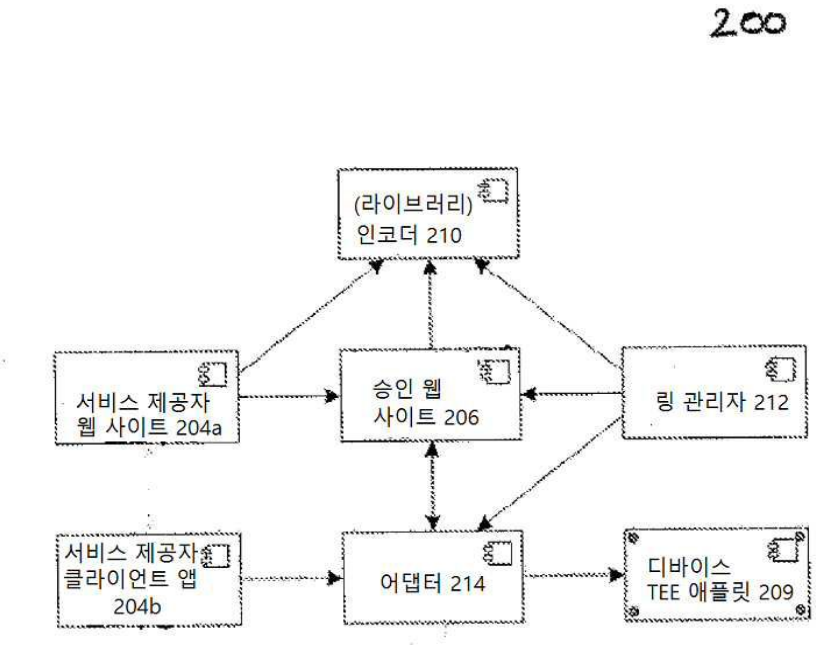


도면2a



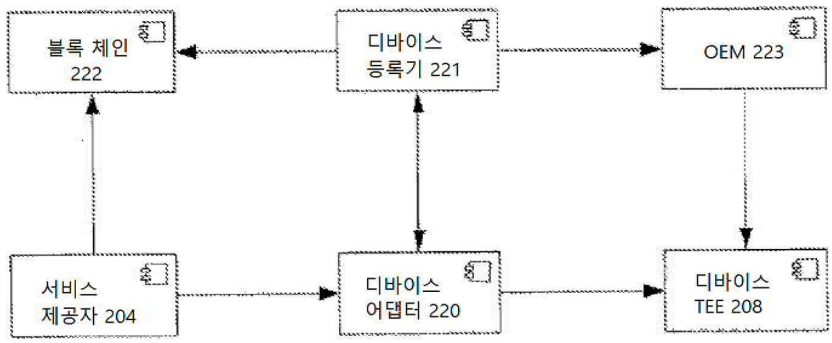
본 발명에 따른 예시적 디바이스 승인 시스템

도면2b



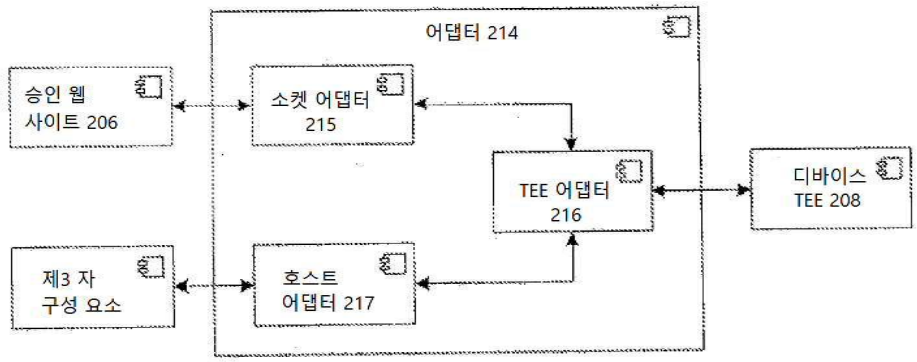
본 발명에 따른 예시적 디바이스 승인 시스템

도면2c



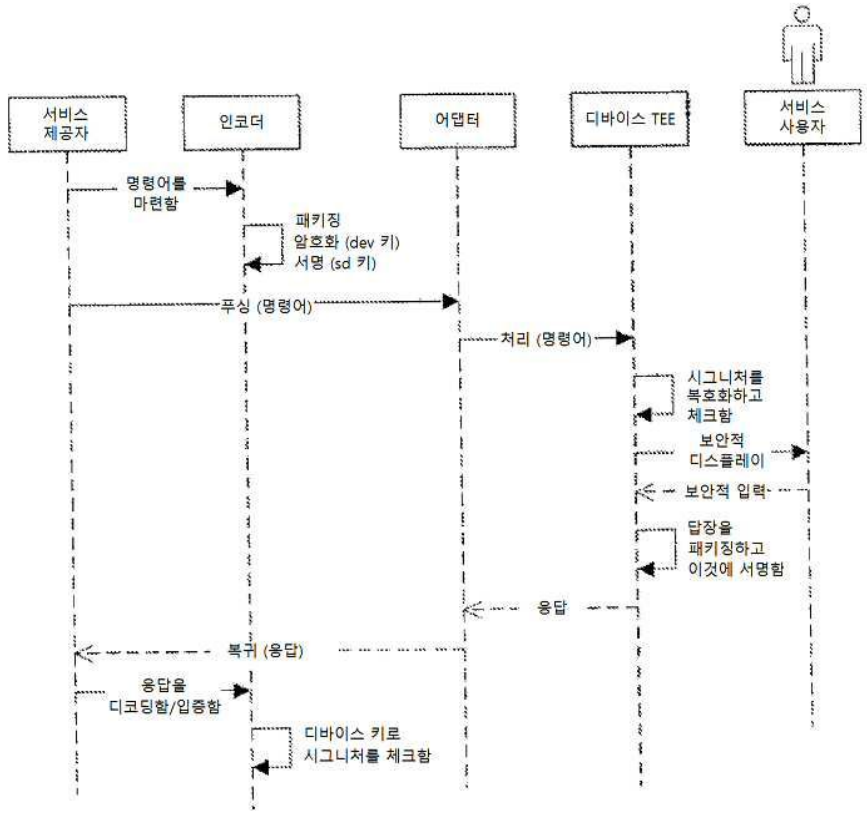
본 발명의 일 실시예의 구성 요소들

도면2d



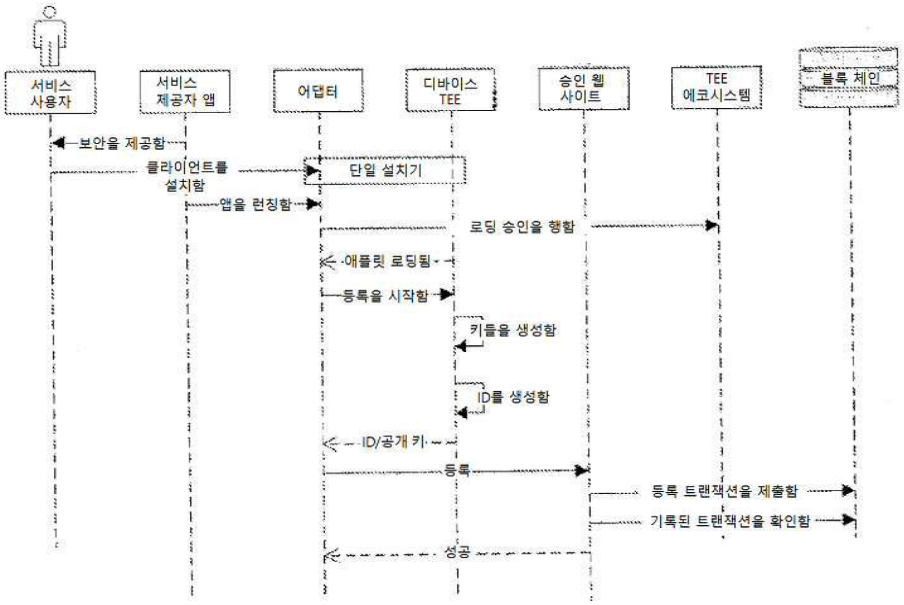
승인 시스템 어댑터 및 승인 시스템 어댑터의 밖으로 그리고 안으로 본 인터페이스들

도면3a



인코더에 의해 명령어를 패키징하고 전달하는 시퀀스

도면3b



본 발명의 일 실시예에 따른 디바이스 등록 프로세스