(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2015년01월02일
(11) 등록번호 10-1477295
(24) 등록일자 2014년12월22일

(51) 국제특허분류(Int. Cl.)
G06F 15/00 (2006.01)
(21) 출원번호 10-2006-0010047
(22) 출원일자 2006년02월02일
심사청구일자 2011년01월26일
(65) 공개번호 10-2006-0097579
(43) 공개일자 2006년09월14일
(30) 우선권주장
11/077,920 2005년03월11일 미국(US)
(56) 선행기술조사문헌
US6605193 B1
US6812965 B1
US6675296 B1

(73) 특허권자
마이크로소프트 코포레이션
미국 워싱턴주 (우편번호 : 98052) 레드몬드 원 마이크로소프트 웨이
(72) 발명자
로즈, 찰스 에프. 3세
미국 98052 워싱턴주 레드몬드 원 마이크로소프트 웨이마이크로소프트 코포레이션 내
코스탈, 그레고리
미국 98052 워싱턴주 레드몬드 원 마이크로소프트 웨이마이크로소프트 코포레이션 내
(뒷면에 계속)
(74) 대리인
제일특허법인

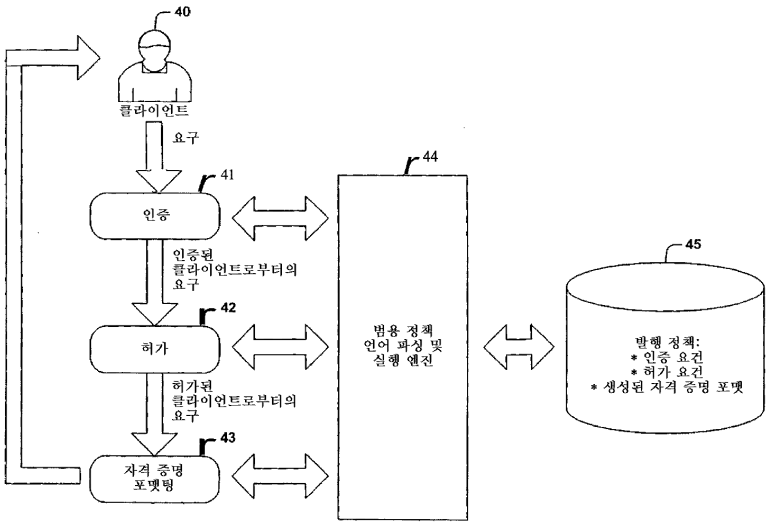
전체 청구항 수 : 총 16 항 심사관 : 홍경아

(54) 발명의 명칭 **인증서 발행을 위한 포맷-독립적 시스템 및 방법**

(57) 요 약

개선된 인증서 발행 시스템은 유입되는 인증서와 인증서 요구를 제 1 포맷에서 제 2 포맷으로 변환하기 위한 인증서 변환 엔진을 포함할 수 있다. 그 후 인증서 발행 엔진은 유입되는 요구에 대해 일반 포맷으로 동작한다. 상기 발행 엔진은 인증서 발행 정책에 따라 인증서를 클라이언트에게 발행할 수 있다. 상기 정책은 실행 시간에 사용될 수 있는 정책 표현 언어로 된 데이터로서 표현될 수 있고, 발행 정책의 유연하고 효율적인 변화를 제공하게 한다. 발행된 인증서는 요구하는 클라이언트에 의해 사용되는 포맷으로 다시 변환될 수 있다. 그러한 변환은 요구하는 클라이언트에게 인증서를 전달하기 전 변환 엔진에 의해 수행될 수 있다.

대 표 도



(72) 발명자

파라마시밤, 무드크리스난

미국 98052 워싱턴주 레드몬드 원 마이크로소프트 웨이마이크로소프트 코포레이션 내

팬다, 라빈드라 나드

미국 98052 워싱턴주 레드몬드 원 마이크로소프트 웨이마이크로소프트 코포레이션 내

코트릴레, 스코트 씨.

미국 98052 워싱턴주 레드몬드 원 마이크로소프트 웨이마이크로소프트 코포레이션 내

라블라, 바산다 케이.

미국 98052 워싱턴주 레드몬드 원 마이크로소프트 웨이마이크로소프트 코포레이션 내

야몰렌코, 블라디미르

미국 98052 워싱턴주 레드몬드 원 마이크로소프트 웨이마이크로소프트 코포레이션 내

종, 유후이

미국 98052 워싱턴주 레드몬드 원 마이크로소프트 웨이마이크로소프트 코포레이션 내

특허청구의 범위

청구항 1

클라이언트 요청에 응답하여 인증서(certificate)를 생성하기 위한 컴퓨터-판독가능 명령어들을 실행하는 적어도 하나의 컴퓨팅 장치를 포함하는 인증서 발행 시스템에 있어서,

프로세서;

클라이언트로부터 제1 포맷의 제1 인증서를 포함하는 상기 클라이언트 요청을 수신하기 위한 수신 컴포넌트 - 상기 수신 컴포넌트는 상기 제1 인증서를 전송함 -;

상기 수신 컴포넌트로부터 상기 제1 인증서를 수신하고, 상기 제1 인증서를 일반 포맷(common format)으로 변환하며, 변환된 제1 인증서를 전송하기 위한 변환 컴포넌트;

상기 변환 컴포넌트로부터 상기 변환된 제1 인증서를 수신하고 인증서 발행 정책을 상기 변환된 제1 인증서로부터의 상기 클라이언트의 정보에 적용함으로써 상기 클라이언트가 제2 인증서를 받을 자격이 있는지 결정하기 위한 발행 컴포넌트 - 상기 인증서 발행 정책은 상기 결정 전에 상기 발행 컴포넌트 상에 로딩되고, 상기 발행 컴포넌트는 상기 클라이언트가 상기 인증서 발행 정책에 의해 요구되는 적어도 하나의 자격 증명을 가지고 있다는 것을 상기 변환된 제1 인증서가 확인하면 상기 일반 포맷의 상기 제2 인증서를 생성하고 상기 제2 인증서를 상기 변환 컴포넌트로 전송하며, 상기 변환 컴포넌트는 상기 제2 인증서를 상기 일반 포맷에서 제2 포맷으로 변환하고 변환된 제2 인증서를 전송함 -; 및

상기 변환된 제2 인증서를 상기 클라이언트에게 전송하기 위한 전송 컴포넌트

를 포함하는 인증서 발행 시스템.

청구항 2

제1항에 있어서,

상기 인증서 발행 정책은 단일의 공통된 정책 표현 언어를 사용하여 표현되는 인증서 발행 시스템.

청구항 3

제1항에 있어서,

상기 인증서 발행 정책은 상기 인증서 발행 정책을 부가 데이터와 결합함으로써 확장되는 인증서 발행 시스템.

청구항 4

제1항에 있어서,

상기 클라이언트는 상기 클라이언트 요청과 함께 수신된 하나 이상의 요구가 상기 클라이언트가 상기 적어도 하나의 자격 증명을 가지고 있음을 증명할 때 인증되는 인증서 발행 시스템.

청구항 5

제1항에 있어서,

상기 클라이언트는 상기 클라이언트 요청과 함께 수신된 하나 이상의 요구가 상기 클라이언트가 상기 적어도 하나의 자격 증명을 가지고 있음을 증명할 때 허가되는 인증서 발행 시스템.

청구항 6

제1항에 있어서,

상기 인증서 발행 정책은 상기 제2 인증서의 형식을 지배하고, 상기 발행 컴포넌트는 상기 형식에 따라 상기 제2 인증서를 생성하는 인증서 발행 시스템.

청구항 7

보안 인증서들을 클라이언트 장치에게 발행하기 위한 방법에 있어서,

인증서 발행 정책을 서버에서 정책 엔진 상에 로딩하는 단계:

상기 서버에서 상기 클라이언트 장치로부터 제2 인증서에 대한 요청을 수신하는 단계 - 상기 요청은 제1 인증서를 포함함 -;

상기 서버에서 상기 제1 인증서를 변환 컴포넌트로 전송하는 단계;

상기 변환 컴포넌트에서, 상기 제1 인증서를 제1 포맷에서 일반 포맷으로 변환하는 단계;

변환된 제1 인증서를 상기 정책 엔진으로 전송하는 단계;

상기 정책 엔진에서, 상기 인증서 발행 정책을 상기 변환된 제1 인증서로부터의 상기 클라이언트의 정보에 적용함으로써 상기 클라이언트 장치가 상기 제2 인증서를 받을 자격이 있는지 결정하는 단계;

상기 클라이언트가 상기 인증서 발행 정책에 의해 요구되는 적어도 하나의 자격 증명을 가지고 있다는 것을 상기 변환된 제1 인증서가 확인하면 상기 일반 포맷의 상기 제2 인증서를 생성하는 단계;

상기 제2 인증서를 상기 변환 컴포넌트로 전송하는 단계;

상기 변환 컴포넌트에서, 상기 제2 인증서를 상기 일반 포맷에서 제2 포맷으로 변환하는 단계; 및

변환된 제2 인증서를 상기 클라이언트로 전송하는 단계

를 포함하는 보안 인증서들을 클라이언트 장치에게 발행하기 위한 방법.

청구항 8

제7항에 있어서,

상기 제2 포맷과 상기 제1 포맷은 동일한 포맷인, 보안 인증서들을 클라이언트 장치에게 발행하기 위한 방법.

청구항 9

제7항에 있어서,

상기 인증서 발행 정책은 상기 인증서 발행 정책을 부가 데이터와 결합함으로써 확장되는 보안 인증서들을 클라이언트 장치에게 발행하기 위한 방법.

청구항 10

제7항에 있어서,

상기 클라이언트는 상기 클라이언트 요청과 함께 수신된 하나 이상의 요구가 상기 클라이언트가 상기 적어도 하나의 자격 증명을 가지고 있음을 증명할 때 인증되는 보안 인증서들을 클라이언트 장치에게 발행하기 위한 방법.

청구항 11

제7항에 있어서,

상기 인증서 발행 정책은 상기 제2 인증서의 형식을 지배하는, 보안 인증서들을 클라이언트 장치에게 발행하기 위한 방법.

청구항 12

클라이언트에게 보안 인증서들을 발행하기 위한 컴퓨터-실행가능 명령어들을 저장하는 컴퓨터 판독가능 저장 매체에 있어서,

상기 컴퓨터-실행가능 명령어들은,

인증서 발행 정책을 정책 엔진 상에 로딩하기 위한 명령어들;

상기 클라이언트로부터 제2 인증서에 대한 요청을 수신하기 위한 명령어들 - 상기 요청은 제1 인증서를

포함함 -;

상기 제1 인증서를 변환 컴포넌트로 전송하기 위한 명령어들;

상기 변환 컴포넌트에서, 상기 제1 인증서를 제1 포맷에서 일반 포맷으로 변환하기 위한 명령어들;

변환된 제1 인증서를 상기 정책 엔진으로 전송하기 위한 명령어들;

상기 정책 엔진에서, 상기 인증서 발행 정책을 상기 변환된 제1 인증서로부터의 상기 클라이언트의 정보에 적용함으로써 상기 클라이언트가 상기 제2 인증서를 받을 자격이 있는지 결정하기 위한 명령어들;

상기 클라이언트가 상기 인증서 발행 정책에 의해 요구되는 자격 증명을 가지고 있다는 것을 상기 변환된 제1 인증서가 확인하면 상기 일반 포맷의 상기 제2 인증서를 생성하기 위한 명령어들;

상기 제2 인증서를 상기 변환 컴포넌트로 전송하기 위한 명령어들;

상기 변환 컴포넌트에서, 상기 제2 인증서를 상기 일반 포맷에서 제2 포맷으로 변환하기 위한 명령어들; 및

변환된 제2 인증서를 상기 클라이언트로 전송하기 위한 명령어들

을 포함하는 컴퓨터 판독가능 저장 매체.

청구항 13

제12항에 있어서,

상기 제2 포맷과 상기 제1 포맷은 동일한 포맷인, 컴퓨터 판독가능 저장 매체.

청구항 14

제12항에 있어서,

상기 인증서 발행 정책은 상기 인증서 발행 정책을 부가 데이터와 결합함으로써 확장되는 컴퓨터 판독가능 저장 매체.

청구항 15

제12항에 있어서,

상기 클라이언트는 상기 클라이언트의 요청과 함께 수신된 하나 이상의 요구가 상기 클라이언트가 상기 자격 증명을 가지고 있음을 증명할 때 인증되는 컴퓨터 판독가능 저장 매체.

청구항 16

제12항에 있어서,

상기 제2 인증서를 생성하기 위한 명령어들은 상기 인증서 발행 정책이 상기 제2 인증서의 형식을 지배하도록 하기 위한 명령어들을 포함하는 컴퓨터 판독가능 저장 매체.

청구항 17

삭제

청구항 18

삭제

청구항 19

삭제

청구항 20

삭제

명 세 서

발명의 상세한 설명

발명의 목적

발명이 속하는 기술 및 그 분야의 종래기술

[0008] 본 출원은 2005년 2월 28일에 출원된 "인증서 발행을 위한 확장성 있는 데이터-구동 시스템 및 방법(Extendable Data-Driven System and Method for Issuing Certificates)"이라는 제목의 미국 특허 출원[변호사 문서 번호. MSFT-4736/311539.01]과 관련되어 있다.

[0009] 본 발명은 컴퓨팅 보안, 그리고 보다 구체적으로는 클라이언트를 서버에 인증하고 클라이언트 허가를 결정하기 위한 디지털 인증서의 사용, 그리고 보다 구체적으로는 그러한 인증서들을 인증서 발행 시스템에서 일반 포맷으로 그리고 일반 포맷으로부터 변환시키는 것과 관련되어있다.

[0010] 인증서는 어떤 것의 진실성이나 소유권을 증명하는 문서이다. 컴퓨팅 세계에서, 디지털 인증서는 다양한 기능을 수행한다. 예를 들어, 디지털 인증서는 엔티티가 그것이 주장하고 있는 것이라는 것을 사실로 확인함으로써 엔티티를 인증할 수 있다. 디지털 인증서는 또한 "정책", 예를 들어, 허가 정책, 신용 정책 등을 부정 방지(tamper-proof) 방식으로 포착할 수 있다.

[0011] 인증서는 매우 유용하고, 현재에는 사용이 증가하고 있다. 보안 정책의 표현과 실행은 점점 더 중요해지는 사업 능력이 되고 있다. 인증서 포맷의 수 역시 급증하고 있다. 오늘날 사용가능한 인기있는 일부 인증서 포맷에는 X.509, 보안 보장 생성 언어(SAML) 보안 토큰, XrML 1.2, 그리고 MPEG-REL이 있다. MPEG-REL에는 XrML 2.x, MPEG ISO-REL, 그리고 ISO-REL을 포함하는 많은 변형과 많은 이름이 알려져 있다는 점을 유의하여야 한다. 약어 MPEG-REL은 여기에서 사용된 것처럼, 적어도 위에 제시된 모든 변형들을 언급하고 있는 것이다.

[0012] 위의 대표적인 인증서의 다양한 기능과 포맷을 기술하면, X.509 인증서는 그들 고유의 포맷을 고수하고 전형적으로 아이덴티티를 표현한다. SAML 인증서는 그들 고유의 XML 스키마를 고수하고 연합 아이덴티티 솔루션들에서 널리 사용된다. XrML 1.2와 MPEG-REL은 자원에 대한 사용 정책을 표현하고 그들 고유의 XML 스키마를 고수한다.

[0013] 오늘날 서비스와 상품들은 인증서를 생산하고 사용하고 있다. 그러나 새로운 타입의 인증서들이 대중화되는 때 문제가 발생한다. 머지않아, 특정 포맷의 인증서를 사용하는 인증서 발행 시스템이 다른 포맷의 인증서와 호환되지 않게 된다. 최선의 경우라도, 클라이언트가 적절히 포맷된 인증서를 얻기 위한 시도를 해야하거나, 또는 클라이언트로 하여금 어떤 인증서 포맷이 서버에 필요한지 미리 결정하게 하는 비효율성을 야기한다. 최악의 경우, 상호 운용성 장애(interoperability failure)의 결과를 낳는다.

[0014] 구현될 수 있는 하나의 가능한 솔루션은 다른 포맷의 인증서를 처리할 수 있는 복수의 협력 인증서 발행 서버를 관리하는 것이다. 이 솔루션은 불행하게도 인증서 발행 시스템의 구현과 업데이트를 더욱 어렵게 만든다. 복수의 시스템을 구현하고 관리하는 데 필요한 노력은 인증서 발행기를 하나씩 추가할 때마다 배가된다.

[0015] 현재 인증서 발행 시스템의 다른 약점은 어떠한 환경하에서 인증서가 발행될 수 있는지, 바꿔 말해서 "인증서 발행 정책"의 환경을 변경하기 어렵다는 점이다. 현재 시스템에서, 상기 정책은 인증서 발행 시스템 이진 코드에서의 컴파일된 알고리즘 또는 구체적으로 모델링 된 구성 파라미터의 "브리틀(brittle)" 셋으로 표현된다. 상기 실행 정책을 고치는 데에는 새로운 인증서 발행 시스템을 재코딩, 재컴파일, 그리고 재배치하는 것이 필요하다. 그러므로, 실제적인 문제로서, 인증서 발행 정책은 인증서 발행 시스템 프로그래머에 의해 미리 예상된 것에 한정된다. 정책을 바꾸기 위해서, 인증서 발행 시스템은 전체적으로 재코딩되어야 할 것이다. 이는 상품 개발 팀의 상당한 양의 시간과 노력을 필요로 한다.

[0016] 그러므로, 인증서 발행 정책의 변화를 용이하게 할 뿐만 아니라, 인증서 발행에 있어서 증가된 상호 운용성을 제공하려는 산업계의 요구가 있었다.

발명이 이루고자 하는 기술적 과제

[0017] 본 발명은 종래 기술의 결점들을 고려하여 개선된 인증서 발행 시스템과 그러한 개선된 시스템에 의해 수행되는 방법들을 제공한다.

[0018] 상기 인증서 발행 시스템은 유입되는 인증서를 일반 포맷으로 변환하고, 출력되는 생성 인증서를 임의의 지원되는 포맷으로 변환하기 위한 변환 구성요소를 포함할 수 있다. 그러므로 상기 시스템은 포맷-독립적이라고 설명될 수 있다. 일반 인증서 발행 정책을 실행하는 단일 인증서 발행 구성요소는 다양한 포맷으로 도착하는 인증서에 대해 동작할 수 있다. 발행 구성요소에 의해 발행되는 인증서들 역시 요청하는 클라이언트들에게 그런 인증서들이 전해지기 전에 다양한 포맷으로 변환될 수 있다.

[0019] 상기 시스템은 또한 인증서 발행 정책을 표현하는 새로운 구성을 포함할 수 있다. 상기 정책은 마크업 정책 표현 언어로 표현될 수 있고 예를 들어 실행 시간에 인증서 발행 시스템에 의해 사용되는 파일에 저장될 수 있다. 따라서 상기 정책은 상기 파일을 바꿈으로써 쉽게 변할 수 있게 된다. 또한 특정한 기술들이 인증서 발행 시스템의 능력을 확장하는데 제공되어 새로운 발행 정책을 적용하고 실행할 수 있게 한다.

[0020] 본 발명의 이점 및 특징은 이하 기술되었다.

발명의 구성 및 작용

[0021] 본 발명의 다양한 실시예의 완벽한 이해를 위해 특정한 특징정 사항들이 이하 설명과 그림들에 상세히 제시되었다. 그러나, 컴퓨팅과 소프트웨어 기술과 관련되어 주지된 일정한 세부 사항들은 본 발명의 다양한 실시예를 불필요하게 모호하게 하는 것을 막기 위해 다음 설명에 있어서 상세히 기술되지 않았다. 또한, 당업자는 이하 기술된 세부사항 중의 일부가 없어도 본 발명의 다른 실시예를 실시할 수 있다는 점을 이해할 것이다. 결국, 다음 설명에 단계 및 순서와 관련하여 다양한 방법들이 설명되어 있어도, 그러한 상기의 설명은 본 발명의 실시예를 명확히 구현하기 위해 제공될 뿐이며, 상기 단계 및 단계의 순서가 본 발명의 실시를 위해 반드시 필요한 것으로 이해되어서는 안 된다.

[0022] 여기에 설명되는 장치와 방법들은 일반적으로 디지털 인증서 발행과 관련된다. "인증서"라는 용어는 여기서 "디지털 인증서"의 줄임말로 사용된다. 종래기술에 언급되어 있듯이, 인증서란 어떤 것의 진실성이나 소유권을 증명하는 문서이다. "증명"이라는 단어는 정확, 진실, 또는 진짜임을 확인하는 것을 의미한다. 그러므로 여기서 클라이언트로 언급될 제 1 엔티티는 인증서를 사용하여 자신의 일부 사실을 제 2 엔티티인 서버에게 확인시킬 수 있다. 상기 인증서는 신뢰받는 제 3자에게서 발행되는 것이 전형적이나, 반드시 그러할 필요는 없다. 여기서 사용된 것처럼, 인증서의 범위는 자가-생성된 문서 또는 토큰에서부터 하나 이상의 공개/개인 키 기술 등에 따른 암호화와 같이, 많은 보안 특징을 갖는 신뢰받는 3자에 의해 발행된 고도의 신뢰받는 디지털 파일에 이른다. 인증서에 의해 증명되는 것은 어느 것이라도 무방하다. 전형적으로, 클라이언트의 아이덴티티 및/또는 일부 자원을 얻거나 액세스하기 위한 클라이언트의 허가가 증명될 수 있으나, 그외 다른 어떤 것도 증명될 수 있다.

[0023] 여기에서 상기 신뢰받는 3자란 인증서 발행 시스템으로 지칭된다. 상기 "인증서 발행 시스템"이라는 용어는 또한 여기 그리고 산업계에서 "인증서 발행 서비스", 그리고 편의를 위해 간단하게 "발행기"라고도 지칭되기도 한다. 인증서 발행 시스템은 어떤 특정 클라이언트가 인증서를 받을 자격이 있는가 결정한다. 만약 자격이 있으면, 클라이언트는 인증서를 발행받아, 서버에 대해 증명하는데 인증서를 사용할 수 있다.

[0024] 클라이언트와 서버를, 하드 드라이브, 버스, 시스템 메모리 등을 포함하는 두 개의 완전한 연산 장치로 생각할 수 있겠지만, 현재 당업자는 이 용어들을 보다 넓은 의미로 이해할 것이다. 사실 클라이언트와 서버는 하나의 연산 장치 내에 있거나, 또는 분산형 연산 장치에서 복수의 컴퓨터에 걸치는 두 엔티티일 수 있다. 이 점에 있어서, 인증서 발행 시스템 또한 하나 이상의 클라이언트 및 하나 이상의 서버 엔티티를 수용하는 컴퓨터 내에 존재할 수 있고, 복수의 장치에 걸쳐서 존재할 수도 있다.

[0025] 이하 도면을 참조하여 본원 발명에 대해 상세히 설명하기로 한다.

[0026] 도 1과 관련하여, 상기 인증서 발행 시스템과 연관하여 사용하기에 적합한 전형적인 연산 장치(100)가 개략적으로 도시되었다. 그것의 가장 기본적인 구성에서, 장치(100)는 전형적으로 프로세싱 유닛(102)과 메모리(103)를 포함한다. 연산 장치의 정확한 구성 및 타입에 따라, 메모리(103)는 휘발성 메모리(103A)(예컨대, RAM), 비휘발성 메모리(103B)(예컨대, ROM, 플래시 메모리), 또는 양자의 어떤 조합일 수 있다. 또한, 장치(100)는 자기 또는 광 디스크 또는 테입과 같은 대용량 저장소(착탈식(104) 및/또는 비-착탈식(105))를 가질 수 있다. 마찬가지로, 장치(100)는 키보드와 마우스와 같은 입력 장치(107), 및/또는 연산 장치(100)의 기능을 액세스하는 그래픽적 도움으로서의 GUI를 나타내는 디스플레이와 같은 출력 장치(106)를 구성요소로 할 수 있다. 장치(100)는 또한 유선 또는 무선 매체를 사용한, 다른 장치들, 컴퓨터, 네트워크, 서버 등과의 통신 연결부(108)를

포함할 수 있다.

[0027] 휘발성 메모리(103A), 비휘발성 메모리(103B), 착탈식 대용량 저장소(104) 및 비-착탈식 대용량 저장소(105)는 컴퓨터 판독 가능 매체의 예시들이다. 컴퓨터 판독 가능 매체에는 컴퓨터 저장 매체 뿐만 아니라, 통신 매체도 포함할 수 있다. 통신 매체는 전형적으로 캐리어 웨이브나 다른 전송 메커니즘과 같은 변조된 데이터 신호에서의 컴퓨터 판독 가능한 명령어, 데이터 구조, 프로그램 모듈 또는 기타 데이터를 포함하고, 임의의 정보 전달 매체도 포함한다. "변조된 데이터 신호"라는 용어는 하나 또는 그 이상의 특성 세트를 갖는 신호로서 그 신호에서 정보를 인코딩하도록 변조된 신호를 의미한다. 통신 매체로는 유선 네트워크 또는 직접-유선 연결과 같은 유선 매체와, 음파, RF, 적외선 및 다른 무선 매체와 같은 무선 매체들이 있는데, 이들에 한정되는 것은 아니다.

[0028] 컴퓨터 저장 매체는 컴퓨터 판독 가능 명령어, 데이터 구조, 프로그램 모듈 또는 기타 데이터와 같이 정보의 저장을 위해 어떠한 방법이나 기술로도 구현될 수 있다. 컴퓨터 저장 매체로는, RAM, ROM, EEPROM, 플래시 메모리나 다른 메모리 기술, CD-ROM, 디지털 다기능 디스크(DVD) 또는 다른 광학 디스크 저장소, 자기 카세트, 자기 테이프, 자기 디스크 저장소 또는 자기 저장 장치, 또는 임의의 기타 매체들이 있는데, 이들에 한정되는 것은 아니다.

[0029] 본 발명은, 적어도 부분적으로는, 프로그램 모듈과 같이 컴퓨터(100)에 의하여 실행되는 컴퓨터-실행가능 명령어를 통해 구현될 수 있다. 일반적으로, 특정 태스크를 수행하거나 특정 추상 데이터 타입을 구현하는 프로그램 모듈은 루틴, 프로그램, 객체, 컴포넌트, 데이터 구조 등을 포함한다.

[0030] 컴퓨터 실행가능 명령어는 일반적으로 컴퓨터 판독 가능한 매체에, 컴퓨터(100)가 이용가능한 디지털 정보로 수록된다. 예를 들어, 도 1에서, 시스템 메모리(103)는 다른 프로그램 모듈 및 프로그램 데이터 뿐만 아니라 운영 시스템 및 어플리케이션 프로그램을 저장할 수 있다. 어플리케이션 등은 운영 체제에 묶일 수도 있고, 또는 별개로 존재하여 운영 체제 서비스가 작용할 수 있도록 유도할 수도 있다.

[0031] 여기서 기술되는 본 발명의 실시예는 소프트웨어 구현이지만, 여기서 기술되는 다양한 기술들은 적어도 일부 프로그램 모듈에 대하여 하드웨어 컴포넌트를 대체함으로써도 구현될 수도 있다는 것을 이해하여야 한다. 그러므로, 현재 본 발명의 방법이나 장치 또는 특정한 형태나 부분들은 고급의 절차 프로그래밍 언어 또는 객체 지향 프로그래밍 언어의 형태를 가질 수 있는 반면, 프로그램(들)은 필요에 따라 어셈블리어나 기계어로도 구현될 수도 있다. 어떤 경우라도, 상기 언어는 컴파일되거나 해석된 언어일 것이며, 하드웨어 구현과 결합되어 있을 것이다.

[0032] 본 발명은 다양한 범용 또는 특수 목적용 컴퓨팅 시스템 환경 또는 구성에서 동작할 수 있다. 본 발명의 사용에 적합한 주지의 연산 시스템, 환경, 및/또는 구성의 예시로서, 개인용 컴퓨터, 서버 컴퓨터, 핸드헬드 또는 랩탑 장치들, 멀티프로세서 시스템, 마이크로프로세서 기반 시스템, 셋탑 박스, 프로그램 가능한 가전, 네트워크 PC, 미니 컴퓨터, 메인프레임 컴퓨터, 상기 시스템이나 장치 등으로 구성된 분산 연산 환경이 있으며 이들에 한정되는 것은 아니다.

[0033] 도 2는 전형적인 네트워크 연산 환경을 나타내고 있다. 상기 네트워크는 객체(273, 274,및 275)뿐만 아니라 연산 장치(271, 272, 276 및 277), 그리고 데이터 베이스(278)를 포함하고 있다. 이러한 엔티티(271, 272, 273, 274, 275, 276, 277 및 278) 각각은 프로그램, 메소드, 데이터 저장소, 프로그램 가능한 논리 등을 포함하거나 이것들을 사용할 수 있다. 상기 엔티티(271, 272, 273, 274, 275, 276, 277 및 278)는 PDA, 오디오/비디오 장치, MP3 플레이어, 개인용 컴퓨터 등의 동일 또는 다른 장치 부분에까지 확장될 수 있다. 각각의 엔티티(271, 272, 273, 274, 275, 276, 277 및 278)는 통신 네트워크(270)를 경유하여 다른 엔티티(271, 272, 273, 274, 275, 276, 277 및 278)와 통신할 수 있다.

[0034] 네트워크의 장치들은 프로토콜 층에 의해 제공되는 기능성을 활용하여 서로 통신한다. 예를 들어, 하이퍼텍스트 전송 프로토콜(HTTP)은 월드 와이드 웹(WWW), 즉 "웹" 과 관련하여 사용되는 일반적인 프로토콜이다. 전형적으로, 인터넷 프로토콜(IP) 주소와 같은 컴퓨터 네트워크 주소 또는 공용 리소스 로케이터(URL)와 같은 다른 레퍼런스들은 서버나 클라이언트 컴퓨터 서로를 확인하는데 사용된다. 상기 네트워크 주소는 URL 주소로 언급될 수 있다. 통신은 통신 수단을 통하여 제공될 수 있는데, 예를 들어, 클라이언트(들)와 서버(들)는 고용량 통신을 위한 TCP/IP 연결(들)을 통하여 서로 결합될 수 있다.

[0035] 상기 네트워크는 그 자체로, 도 2의 시스템에 서비스를 제공하는 다른 연산 구성요소를 포함할 수 있으며, 그 자체로 복수의 상호 접속된 네트워크를 표현할 수 있다. 각각의 엔티티(271, 272, 273, 274, 275, 276, 277

및 278)는 하나 또는 그 이상의 다른 엔티티(271, 272, 273, 274, 275, 276, 277 및 278)의 서비스를 요청하기 위해서, API 또는 다른 객체, 소프트웨어, 펌웨어 및/또는 하드웨어를 사용할 수 있는 별개의 기능 프로그램 모듈을 포함할 수 있다.

[0036] "클라이언트"는 클래스나 그룹의 일원으로서, 연관되지 아니한 다른 클래스나 그룹의 서비스를 사용한다. 연산에 있어서, 클라이언트는 다른 프로그램에서 제공되는 서비스를 요청하는 프로세스, 바꿔 말하면, 대개 명령어나 태스크 세트가 될 수 있다. 예를 들어, 이러한 서비스에는 인증서 발행 시스템에 의한 인증서의 발행을 들 수 있다. 상기 클라이언트 프로세스는 다른 프로그램이나 서비스 그 자체에 관한 어떠한 작업 세부사항을 알 필요 없이 요청된 서비스를 활용할 수 있다. 클라이언트/서버 아키텍쳐에서, 특히 네트워크 시스템에서, 클라이언트는 다른 컴퓨터, 예를 들어 서버에 의해 제공되는 공유 네트워크 자원에 액세스하는 컴퓨터인 것이 보통이다. 도 2의 예에서, 어떠한 엔티티(271, 272, 273, 274, 275, 276, 277 및 278)도 환경에 따라서 클라이언트, 서버, 또는 클라이언트와 서버 모두로 생각될 수 있다.

[0037] 반드시 필연적으로 그러한 것은 아니지만, 서버는 전형적으로, 인터넷처럼 원격 또는 로컬 네트워크로 접속가능한 원격 컴퓨터 시스템이다. 클라이언트 프로세스는 제 1 컴퓨터 시스템에서 활성화될 수 있고, 서버 프로세스는 통신 매체를 통하여 서로 통신하면서 제 2 컴퓨터 시스템에서 활성화될 수 있으므로, 분산된 기능성을 제공하고 복수의 클라이언트가 서버의 정보 수집 능력을 이용할 수 있게 할 수 있다. 어떤 소프트웨어 객체도 복수의 연산 장치 또는 객체에 걸쳐서 분배될 수 있다.

[0038] 그러므로 본 발명의 실시예는 인증서를 요구하는 클라이언트 엔티티가 네트워크의 제 1 연산 장치(예를 들어 277)에 존재하고 있는 상황을 언급하고 있는 것이다. 클라이언트 엔티티가 필요로 하는 일부 자원을 가지고 있을 서버 엔티티는 제 2 연산 장치(예를 들어 275)에 존재한다. 인증서 발행 시스템은 제 3 연산 장치(예를 들어 271)에 존재할 수 있다.

[0039] 제 1 연산 장치(277)에 있는 클라이언트는, 제 2 연산 장치(275)에 있는 서버를 위한 일부 클라이언트 자격 증명(들)을 증명할 인증서를 필요로 하는지 결정할 수 있다. 그러므로 제 1 연산장치(277)에 있는 클라이언트는 네트워크 버스(270)를 통하여 제 2 연산 장치(271)에 있는 발행기에게 요청을 한다. 상기 요청은 그 자체로 하나 또는 그 이상의 예전 발행되었던 인증서를 포함할 수 있다. 제 3 연산 장치(271)에 있는 발행기는 클라이언트가 요청한 인증서를 받을 자격이 있는가 결정하는 작업을 진행한다. 제 3 연산 장치(271)에 있는 발행기는 인증서 발행 정책을 적용함으로써 이 작업을 수행한다. 만약 제 1 연산 장치(277)에 있는 클라이언트가 정책에 의해 요구되는 자격 증명을 가지고 있는 경우, 요청된 인증서는 제 3 연산 장치(271)에 있는 발행기에 의하여 제 1 연산 장치(277)에 있는 클라이언트에게 발행될 수 있다. 그 후, 클라이언트는 발행된 인증서를, 임의의 수의 다른 인증서들과 함께, 제 2 연산 장치(275)에 있는 서버와의 통신에 사용하게 된다.

[0040] 도 3a는 인증서 발행 시스템의 변환 컴포넌트 부분을 나타낸다. 발행 컴포넌트(34)는 도 4 및 5에서 보다 자세하게 나타내었다. 본 발명의 다양한 실시예에서, 상기 변환 컴포넌트는 실질상, 들어오는 인증서를 수신하는 제 1 컴포넌트가 될 수 있고, 또한 클라이언트에게 전달되기 전에 인증서를 조작하는 마지막 컴포넌트가 될 수 있다. 다른 실시예에서, 필요에 따라서 인증서 발행 시스템의 임의의 위치에 변환 컴포넌트를 사용할 수 있다.

[0041] 인증서 요청이 인증서 발행 시스템에 도착할 때, 제 1 포맷(30)에서의 제 1 인증서와 같이, 임의의 수반되는 인증서들도 변환 드라이버(31)로 보내질 수 있다. 변환 드라이버(31)는 발행 컴포넌트(34)의 동작에 사용되는 일반적 포맷으로 인증서를 변환하거나 일반적 포맷으로부터 인증서를 변환할 수 있도록 동작한다. 그러므로, 제 1 포맷(30)에서의 제 1 인증서는 제 2 포맷으로 변환될 수 있다. 그러한 변환의 결과는 제 2 포맷(33)에서의 제 1 인증서이다.

[0042] 이와 반대로, 인증서가 발행 컴포넌트(34)에 의해 발행될 때, 인증서는 발행 컴포넌트(34)에 의해 생성된 포맷으로부터 클라이언트가 요구하는 어떤 포맷으로도 변환될 수 있다. 그러므로, 제 2 포맷(35)에서의 제 2 인증서는 제 3 포맷으로 변환될 수 있다. 그러한 변환의 결과는 제 3 포맷(37)에서의 제 2 인증서이다. 이 제 3 포맷은 클라이언트의 인증서 요구와 함께 도착하는 하나 또는 그 이상의 인증서 포맷(예를 들어 30)을 포함하는 어떠한 인증서 포맷도 될 수 있다.

[0043] 상기 인증서 변환 컴포넌트는 되도록 많은 인증서 포맷을 변환하기 위해서 최적으로 설계되어있다. 이 점에 있어서, 인증서 변환 컴포넌트는 X.509 인증서, 보안 보장 생성 언어(SAML) 보안 토큰 인증서, XrML 1.2 인증서, 그리고 MPEG-REL 인증서를 몇몇 더욱 유명한 전형적 인증서 포맷으로 변환할 수 있다. 그러나, 각각 그리고 모

든 가능한 인증서 타입에 대해서 변환 장치를 설계하는 것이 경제적으로 볼 때 항상 가능한 일은 아니다. 본 발명은 변환 컴포넌트에 의해 변환 가능한 인증서 타입의 특정 포맷이나 수에 한정되지 않는다.

[0044] 새로운 인증서 포맷이 본 기술분야에서 계속해서 생성되기 때문에, 변환 컴포넌트가 부가적인 인증서 포맷을 수용할 수 있도록 확장 가능하도록 설계하는 것이 유리하다. 드라이버(31)는 하나 또는 그 이상의 클래스, 예를 들어 변환 클래스(32)에 의해 제공되는 명령어에 기초하여, 특정한 인증서(30)의 다양한 요소를 일반 포맷 인증서(33)로 변환할 수 있다. 도 3a의 특정한 배치로 인해 새로운 인증서 포맷을 수용할 수 있게 되므로 변환 컴포넌트의 확장성에 있어서 유리하다. 그러나, 도 3a에서 드라이버(31) 및 변환 클래스(32, 36)와 같은 클래스의 배치는 단지 예시적인 것일 뿐이며, 당업자라면 인증서를 한 포맷에서 다른 포맷으로 변환시키기 위해 다양한 배치들이 구현될 수 있다는 점을 인지할 것이다.

[0045] 도 3a에서, 일반 포맷은 제 2 포맷이라고 언급되어 있다. 일반 포맷의 선택은, 유입되는 다양한 포맷의 인증서에 포함될 다양한 타입의 정보를 모두 수용할 수 있게 하기 위해, 가능한 한 견고한 포맷을 결정하는 작업을 포함한다. 어떠한 인증서도 소위 더욱 유명한 전형적 인증서 포맷인, X.509 포맷, 보안 보장 생성 언어(SAML) 보안 토큰 포맷, XrML 1.2 인증서, 및 MPEG-REL 포맷을 포함하는 일반 포맷으로서 수행되도록 선택될 수 있는데, MPEG-REL이 본 발명의 구현을 위해 선택되어 왔고 현재 이 목적에 유리한 것으로 생각된다. MPEG-REL이 현재 가장 선명하고, 강하며 확장성 있는 인증서 포맷으로 생각된다. 그러나, 기술이 발전함에 따라 다른 더 좋은 인증서들도 개발될 것이다.

[0046] 변환 컴포넌트 사용의 다른 이점은 다양한 인증서 포맷 각각이 자신 고유의 방법으로 정책을 표현한다는 사실에서 유래한다. 현재 인증서 발행기의 상호 운용성에 있어서의 장애물은 포맷 부호환성인데, 왜냐하면 어떤 특정 포맷을 쓰기 위해서는 발행기를 투과하는 커스텀 알고리즘이 필요하기 때문이다. 우리는 존재하고 있는 모든 인증서 발행자가 일반 포맷을 채택할 것이라고 확신할 수는 없으므로, 이러한 포맷들이 오랜 기간 동안 계속해서 존재할 것이라고 가정해야만 한다. 따라서 별개의 인증서 포맷과 그 의미를 일반언어로 매핑하거나 변환하는 기술은 복수 포맷의 시스템적 충격을 줄이고, 인증서 상호 운용성의 문제를 해결하는 방향으로의 나아가게 된다.

[0047] 정확한 변환이 되기 위해선 문법과 어의 요건 모두를 만족시켜야만 한다. 전자는 변환된 인증서가 유효한 포맷을 가지고 있어야 함을 요구하는 것이다. 후자는 상기 변환된 인증서가 원본 인증서와 동일한 정보를 가지고 있어야 함을 요구한다. 그러나 소스 포맷이 목적 포맷보다 더 많은 정보를 가지게 되어 변환 과정에서 불가피하게 정보 손실이 생기는 경우가 있다. 그러므로 본 발명의 구현 목표는 최선을 다하여 다른 정보를 보존하는 한편 정보를 정확하게 변환하는 것을 보장하는 것이다.

[0048] 인증서 변환 알고리즘(32, 26)은 그들의 기능에 기초하여 두 개의 카테고리로 범주화될 수 있다.

[0049] 문법 레벨(Syntax level): 컨스트럭트 간 매핑

[0050] 어의 레벨(Semantic level): 인증서 간 변환

[0051]

[0052] 한 실시예에서, 이러한 알고리즘의 모임은 클래스 세트로 구현된다. 클래스 세트는 도 3a에 인증서 변환 드라이버(31)로 나타나 있다. 변환 드라이버(31)의 실시예는 세 구성요소로 구성될 수 있다: 드라이버 클래스, 인증서 변환 클래스 및 시스템 구성 클래스. 상기 드라이버 클래스는 대체적인 변환 프로세스의 조정을 수행하고 변환 클래스를 호출한다. 상기 변환 클래스는 실제의 변환 프로세스를 수행할 수 있고, 시스템 구성 클래스는 변환 드라이버(31)가 동작하는데 필요한 구성 데이터를 보유할 수 있다.

[0053] 뒤의 간단한 예시는 제 1 인증서 포맷, XrML 1.2로부터 제 2 인증서 포맷, XRML 2.0으로 변환하는 전형적인 동작을 보여주기 위한 것이다. 당업자라면 이해할 수 있듯이, MICROSOFT® Rights Management Server 제품은 정책 평가를 하기 위해서 XrML 1.2 포맷으로 되어있는 인증서를 사용했다. 그러나, 미래에 계획된 Rights Management Server 제품의 발매물은 XrML 2.0 포맷으로 된 인증서를 사용하게 될 것이다. 그러므로, 이하 내용들은 변환 컴포넌트의 동작과, 여기서 제공되는 본 발명에 대한 증가하고 있는 요구의 좋은 예시가 된다. 상황은 다음과 같다: 클라이언트가 본 발명을 구현하고 있는 가상의 Rights Management Server product에 인증서 요청을 보낸다. 상기 요청은 그 자체에 XrML 1.2 포맷으로 된 인증서, 예를 들어 도 3a의 인증서(30)를 포함한다. XrML 2.0은 인증서 발행 시스템(34)에 의해 사용되는 일반 포맷이다.

[0054] 두 포맷으로 된 인증서 모두를 수신하고 이해하기 위해서, 변환 클래스의 세트가 구현된다. 이것이 이루어진다고 가정할 때, XrML 1.2 포맷의 인증서(30)를 가진 클라이언트로부터의 요구가 도착하자마자, 변환 드라이버(31)는 이에 대응하는 변환 클래스(32)를 생성하고, 구성 데이터를 그것에 전달하며 변환 메소드를 불러낸다. 인증서 변환 클래스(32)는 XrML 1.2 인증서(30)의 서명 및 유효 간격을 유효화하고, 그것을 XrML 2.0 상응 부분(33)으로 변환하고, 변환-특정 정보를 변환 드라이버(31)로 반환한다. 그러면 변환 드라이버(31)는 XrML 2.0 포맷의 인증서(33)를 사용하는 Rights Management Server(34)를 호출한다. 이 예시에서, Rights Management Server(34)는 XrML 2.0 인증서를 이해하고 인증서 발행 동작을 수행하고 그 결과를 변환 드라이버(31)로 돌려준다. Rights Management Server(34)에 의해 발행된 XrML 2.0 인증서(35)를 받은 후에, 변환 드라이버(31)는 인증서 변환 클래스(36)를 생성하여 상기 XrML 2.0 인증서(35)를 XrML 1.2 대응부분(37)으로 변환하게 된다.

[0055] 도 3b는 들어오는 인증서 요구가 발행기에 의하여 어떻게 다루어지는지 나타내었다. 처음으로 요구가 서버 엔트리 포인트(301)를 히트한다. 자신의 독자적인 포맷으로 되어있는 상기 인증서 또는 다른 요구 정보(예를 들어 302)는, 인증서 변환 층(303)이라 불리는 변환 컴포넌트에 전달된다. 인증서는 항상 인증서 요구와 함께하는 것은 아님에 유의하여야 한다. 일부 형태에서 증명이 이미 이루어졌고, 인증서가 그 사실을 증명할 수 있는 유리한 방법이라는 것이지, 항상 모든 인증서 요구와 함께하여야 하는 것은 아니다. 다양한 종류의 다른 정보들 역시 요구에 포함될 수 있다. 일부 다른 정보가 사용되는 경우, 정보의 특정 데이터 종류은 인증서 포맷이 다를 수 있는 것처럼 다소 다를 수 있다. 그러므로, 인증서 변환 층(303)은 제 1 인증서 포맷(302)과 같은 인증서를 한 포맷에서 다른 포맷으로, 또는 다양한 포맷에서 단일 포맷으로 변환하도록 구성될 수 있고 또한 인증서 요구와 함께하는 어떤 데이터를 다루도록 구성될 수 있다. 그러한 데이터는, 들어오는 인증서처럼, 일반 포맷으로 변환되어 단일 발행 컴포넌트 또는, 여기서 언급된 바와 같이, 인증서 엔진(305)에 의해 사용될 수 있다.

[0056] 도 3b의 유입되는 정보는 제 2 포맷(304)으로 변환될 수 있다. 만일 유입되는 정보가 이미 제 2 포맷인 경우, 자연적으로 변환이 필요하지 않게 된다. 요구(304)와 연관된 인증서 및/또는 다른 정보는 그 다음으로 서버 인증서 엔진(305)에 의해 다루어진다.

[0057] 도 3c는 클라이언트(310)가 서버(313)에 인증서를 요구하는 본 발명의 다른 형태를 나타낸다. 타입 1 인증서는 클라이언트(310) 요구와 연관되어 있다. 타입 1 인증서는 인증서 변환 엔진(CTE)(311)에 의하여 발행기(312)가 사용할 수 있는 타입 2 인증서 등의 다른 포맷으로 변환될 수 있다. 상기 타입 2 인증서는 발행 엔진(312)에 의해 사용되어 클라이언트(310)의 요구를 승낙할지 결정할 수 있다. 발행 엔진(312)가 제 2 포맷으로 인증서를 발행할 때, CTE(311)는 클라이언트(310)를 위하여 그것을 제 1 포맷으로 변환할 수 있다. 이와 달리, 발행 엔진(312)은 제 2 포맷 인증서를 제 1 포맷 인증서로 변환하도록 CTE(311)를 재사용하는 것 없이, 클라이언트(310)를 위하여 적절한 포맷으로 인증서를 간단하게 생성할 수 있다. 본 발명의 많은 실시예에서, CTE(311)는 복수의 유입 인증서 타입을 다룰 수 있도록 구성될 수 있다는 것에 유의하여야 한다. 각각의 그러한 타입은 되도록이면 발행기(312)의 동작을 위해 일반 타입으로 변환된다. 반대로, CTE(311)는, 클라이언트(310)에 의해 요구된 바와 같이, 되도록 발행 엔진(312)으로부터의 제 2 포맷 인증서를 임의의 다양한 인증서 타입으로 변환하도록 구성된다.

[0058] 전형적인 시나리오들 중에서, 도 3c에 표시된 것처럼 시스템을 사용하여 이득을 얻을 수 있는 것에는 클라이언트 장치 및/또는 프로세스의 인증, 클라이언트 허가자 인증서(CLC;Client Licensor Certificate) 요구의 허여, 그리고 디지털 콘텐츠에서 권리의 허가가 있다.

[0059] 전형적인 인증 시나리오가 발생하는 때는 제 1 포맷의 인증서로 동작하는 클라이언트("v1 클라이언트")를 사용하는 사용자가 권리 관리 시스템에 의해 보호되는 이메일을 처음으로 열기 위한 시도를 할 때이다. 이 시나리오에서, 예를 들어 v1 클라이언트는, 현재 클라이언트 머신의 아이덴티티를 나타내는 XrML 1.2 인증서인 v1 Machine Account Certificate(MAC) 및 WINDOWS® 도메인 자격 증명을 인증서 발행 시스템(313)에 보낼 수 있다. CTE(311)는 상기 MAC을 현재 클라이언트 머신의 아이덴티티를 나타내는 MPEG-REL 인증서인 Security Processor Certificate(SPC)로 변환할 수 있고, 상기 SPC를 발행 엔진(312)에 줄 수 있다. 그 다음 상기 발행 엔진은 제 2 포맷 인증서, 예를 들어 현재 클라이언트 머신에서 사용자의 아이덴티티를 나타내는 MPEG-REL 인증서인 Right Account Certificate(RAC)를 발행하고, RAC를 CTE(312)으로 보낼 수 있다. 그 다음 CTE(311)는 RAC를 제 1 포맷 인증서, 예를 들어 현재 클라이언트 머신에서 사용자의 아이덴티티를 나타내는 XrML 1.2 인증서인 Group Identity Certificate(GIC)로 변환할 수 있다. 그 다음 발행 시스템(313)은 GIC에 의해 클라이언트(310)에 응답할 수 있다.

[0060] 전형적인 CLC 요구 시나리오는 클라이언트(310)가 발행 시스템(313)에 요구를 보내서, 사용자로 하여금 보호되는 콘텐츠에 대한 라이센스를 오프라인으로 발행하도록 허가하는 인증서를 얻는 것이다. 이 경우에, CTE(311)는 그 요구와 연관된 유입 RAC를 발행 엔진(312)에 의해 처리될 수 있는 CLC로 변환할 수 있다. 이와 달리, 클라이언트(310)는 CTE(311)에 의해 변환된 GIC를 SPC로 보낼 수 있고, 발행기(312)는 CTE(311)에 의해 변환된 RAC를 CLC로 반환할 수 있다.

[0061] 디지털 콘텐츠에 있어서 권리의 허가에 사용되는 도 3c와 같은 시스템의 전형적인 시나리오는 다음과 같다: 권리-보호된 콘텐츠를 사용하기 위해서, v1 클라이언트(310)는 발행 시스템(313)에 허가 요구를 할 수 있다. 클라이언트(310)는 v1 발행 허가 Issuance License(IL), 예를 들어 보호 콘텐츠의 특정 부분의 저작자 지정된 사용 권리를 기술하는 XrML 1.2 인증서와, GIC를 발행 시스템(313)에 보낼 수 있다. 그 다음 CTE(311)는 들어오는 IL과 RAC를 변환하여, 발행 엔진(312)에 줄 수 있다. 만일 상기 발행 엔진(312)이 Use License(UL), 예를 들어 보호되는 콘텐츠의 특정 부분에 특정 사용자가 액세스할 수 있도록 허가하는 MPEG-REL 인증서를 클라이언트(310)에 수여하면, CTE(311)는 또한 발행된 UL을, v1 클라이언트(310)에 의해 사용되는, 보호된 콘텐츠의 특정 부분에 특정 사용자가 액세스할 수 있도록 허가하는 XrML 1.2 인증서와 같은 End User License(EUL)로 변환될 수 있다. 그 다음 상기 발행 시스템(313)은 UL을 클라이언트(310)로 돌려보낼 수 있다.

[0062] 당업자라면 이해할 수 있듯이, CTE(311)을 구현하기 위하여 많은 소프트웨어 디자인들이 사용되었는데, 전형적인 CTE 설계는 세 컴포넌트를 구성요소로 할 수 있다: CTE 드라이버, 인증서 변환 클래스, 및 구성 클래스.

[0063] 이 설계에서, CTE 드라이버는 서버 엔트리 포인트(301) 및 서버 인증서 엔진(305)과 상호 작용한다. 인증서 요구를 수신하자마자, 상기 드라이버는 대응하는 인증서 변환 클래스를 생성하고, 구성 데이터를 변환 클래스에 전달하고, 변환 메소드를 불러낸다. 그 다음 인증서 변환 클래스는 취급하고 있는 인증서의 서명 및 유효 간격을 유효화하는데, 예를 들어 a) 서명 유효화, b) 인증서 유효 기간 만료 시간 및 c) 발행기를 신뢰된 발행기 세트와 비교하기와 같은 기술들을 사용한다. 상기 인증서 변환 클래스는 또한 인증서를 상응하는 일반 포맷 인증서 변환하고, 일반 포맷 인증서 스트링을 다른 변환-특정 정보와 함께 CTE 드라이버에 반환한다.

[0064] CTE 드라이버는 다음으로 인증서 엔진(305)을 일반 포맷 인증서를 사용하여 호출한다. 인증서 엔진(305)에 의해 발행된 생성된 일반 포맷 인증서를 받은 후, CTE 드라이버는 인증서 변환 클래스를 생성하여 생성된 일반 포맷 인증서를 상응하는 제 3 포맷, 예를 들면 원래의 유입 인증서의 포맷의 인증서로 변환한다.

[0065] 그러므로, CTE가 진행할 수 있는 전형적인 작업 흐름은 다음과 같다:

[0066] 클라이언트로부터 인증서 요구를 받음

[0067] 수반하는 인증서를 유효화하고 필요한 경우 해독함

[0068] 해독된 인증서를 상응하는 일반 포맷으로 변환함

[0069] 특정 정보 (키, SPC 등)를 대체할 수 있음

[0070] 일반 포맷 인증서에 표시되지 못하는 임의의 정보를 저장함

[0071] 인증서 발행 엔진을 호출함

[0072] 인증서 발행 엔진으로부터 일반 포맷 인증서를 발행받음

[0073] 필요한 경우 상기 발행된 일반 포맷 인증서를 해독함

[0074] 해독된 발행된 일반 포맷 인증서를 클라이언트-포맷의 상응 부분으로

[0075] 변환함

[0076] 이전 스텝에서 저장된 상기 정보를 사용할 수 있음

[0077] 클라이언트-포맷의 인증서를 암호화하고 서명함

[0078] 클라이언트 포맷의 인증서를 클라이언트에게 보냄

[0079] 도 4는 인증서 발행 시스템의 발행 컴포넌트 부분을 나타낸다. 도 3a와 관련하여 기술된 것처럼 변환이 일단 완

료되면, 변환된 인증서는 도 4에 나타난 다양한 기능적 컴포넌트를 사용하여 처리될 수 있다. 상기 변환 컴포넌트는 도면의 간략화를 위해서 도 4나 도 5에 나타내지 않았다. 다양한 컴포넌트의 전형적인 집적을 알아보기 위해서는 도 3a를 참조하라.

[0080] 클라이언트(40)는 하나 또는 그 이상의 요청을 인증서 요구와 함께 보낸다. 요청은 엔티티에 의해 수행되는 것으로서 엔티티가 인증서를 받을 자격이 있는지 없는지 결정하는데 사용된다. 요청이 진실한 것으로 증명되면, 클라이언트 엔티티(40)는 자격 증명을 가지고 있다는 것이 증명된다. 클라이언트(40)로의 인증서 발행에 앞서 궁극적으로는 하나 이상의 자격 증명이 인증서 발행 정책에 의해 필요하게 된다. 그러한 자격 증명들은 하나 이상의 인증서를 통해 그들 자신을 증명받을 수 있다.

[0081] 인증(41), 허가(42), 및 자격 증명 포맷팅(43)은 발행 컴포넌트에 의해 수행되는 전형적인 기능들이다. 도 4에서 나타낸 것처럼, 이러한 기능들은 인증(41), 허가(42), 자격 증명 포맷팅(43)의 순서로 연속적으로 수행될 수 있다. 이런 순서는 모든 실시예에서 요구되는 것은 아니다. 클라이언트 요구를 만족시키기 위해 무엇이 필요한가에 따라서, 인증(41), 허가(42) 및 자격 증명 포맷팅(43)은 각각 독립적으로 수행될 수 있고, 일부 서브-콤비네이션으로 수행될 수 있고, 또는 하나 이상의 상기 기능들이 도 4에 기술되지 아니한 다른 기능들과 함께 수행될 수도 있다.

[0082] 인증(41), 허가(42) 및 자격 증명 포맷팅(43)이 연속적으로 수행될 때, 인증 프로세스(41)는 인증서 요구를 지원하라는 요청을 하고 있는 클라이언트(40)가, 진실로 클라이언트(40) 요청이 요구하고 있는 엔티티인지, 처음으로 결정하게 된다. 이것이 증명되면, 그 다음으로 허가 프로세스(42)가 클라이언트가 요구한 인증서를 받을 수 있도록 허가되었는지 결정한다. 이와 달리, 허가 프로세스(42)는 간단히 클라이언트(40)의 허가 레벨을 결정하고 생성되는 인증서에 그것을 기록할 수도 있다. 마지막으로, 인증서가 클라이언트(40)를 위하여 생성될 때, 인증서에 수록된 클라이언트(40) 자격 증명은 생성된 인증서 내에서 자격 증명 포맷팅(43)에 의해 포맷팅된다. 인증서 발행 엔진(44)에 의해 적용되는 전형적 알고리즘을 다음에 나타내었다.

[0083] IssueCertificate(User input certificates, Server Issuance Policy)

[0084] Based on issuance policy

[0085] Authenticate user input certificates

[0086] Authorize server for issuing certificate

[0087] Construct resultant certificate as follows

[0088] Create what client engine needs to authenticate

[0089] Create grants allowed by issuance policy

[0090] Authorize grant issuance

[0091] Call extension to authorize if required

[0092] Generate grant

[0093] Call extension to generate part of the grant if required

[0094] Construct user rights as grants

[0095] Sign the generated certificate

[0096] Return the certificate

[0097] 인증(41), 허가(42), 및 자격 증명 포맷팅(43) 기능을 수행하는 부분에서, 범용 정책 언어 파싱 및 실행 엔진(44)은 인증서 발행 정책(45)을 적용할 수 있다. 실행될 정책이 엔진(44)에서 표시될 필요가 없고, 실행시에 엔진(44)에 의해 사용되는 발행 정책(45) 내에 표시되면 된다는 점에서 볼 때에, 상기 발행 구성요소는 데이터-구동이라 할 수 있다.

[0098] 이전 기술 인증서 발행 시스템은 인증서를 생성할 때 정책을 적용하고 실행하는데 반해, 이 정책은 이전의 기

술 발행기에서 인증서 발행 시스템 이진 코드에서의 컴파일된 알고리즘 또는 구체적으로 모델링 된 구성 파라미터의 "브리틀" 세트로 표현된다. 그 결과, 이전의 기술 발행기에서 실행 정책을 바꾸는 데에는 새로운 발행기 바이너리를 재코딩, 재컴파일, 그리고 재배치하는 것이 필요하다. 다르게 말하면, 전달된 발행기는 인증서 발행 시스템 프로그래머에 의해 미리 예상된 정책들의 세트를 실행하는 것에 한정된다.

[0099] 인증서 발행 엔진(44)은 미리 예상된 정책 구조를 거의 포함하지 않거나 가지고 있지 않아야 한다. 대신에, 엔진(44)은 실행시간에 만나는 실행 정책(45)으로부터의 특정 정책 데이터를 취급하는 메타-데이터 구동 정책 실행 엔진을 포함한다. 상기 정책 데이터(45)는 엔진(44)이 사용할 수 있도록 설계된 범용 확장 정책 표현 언어를 사용하여 표현되어 있다.

[0100] 엔진(44)은 단일의 일반 정책 표현 언어에서 동작하고, 실행 정책(45)에 있는 사용 가능 정책과 데이터에 기초하여 허가 결정을 내리는 것이 바람직하다. 상기 프로세스를 균질의 정책 표현 언어 포맷으로 수행함으로써, 엔진(44)의 논리는 더욱 간단하고, 효율적이며, 선택된 정책 표현 언어에 최적화될 수 있다. 데이터-구동형이 됨으로써, 엔진(44)은 새로운 정책, 어의 또는 구조를 수용하기 위한 논리를 변화시킬 필요 없이, 광범위한 표현 정책들을 평가할 수 있게 된다. 도면에 표시된 바와 같이 엔진(44)은 정책(45)을 파싱 및 실행하기 위한 기능적 컴포넌트와, 인증서 생성을 위한 기능적 컴포넌트 모두를 포함하고 있다. 엔진(44)에 의해 생성된 인증서의 형식은 정책(45)과 발행 정책의 다른 형태에 의해 지배될 수 있다.

[0101] 정책(45)을 표현하기 위하여 사용된 정책 표현 언어는 넓고 다양한 형식을 취할 수 있다. 사용된 언어는 XML(Extensible Markup Languate), HTML(Hyper Text Markup Language), 또는 일부 다른 마크업 언어가 될 수 있다. 당업자라면 이해할 수 있듯이, 인간이 읽을 수 있는 단어와 부호의 세트는 의도하는 동작을 정확히 명시하기 위한 언어들로 조합될 수 있다. 엔진(44)과 같은 머신 프로세스는 실행 시간에 상기 형식으로 된 파일을 사용하고 의도한 동작을 수행하도록 구성될 수 있다. 본 발명에 사용하도록 설계된 어떤 정책 표현 언어도 견고하고, 확장성이 있으며 필요한 때 정책의 변화와 언어 의미의 부가를 수용할 수 있도록 유연하여야 한다. 마크업이란 문서의 논리적 구조를 기술하기 위하여 텍스트나 워드 프로세싱 파일의 특정 장소에 삽입되는 문자 또는 다른 심벌의 순서를 말한다. 상기 마크업 지시기는 종종 "태그" 라고 불린다. 마크업은 문서 작성자에 의해 직접적으로 심벌들을 타이핑함으로써 또는 에디터를 사용하여 미리 패키지된 마크업 심벌들을 선택함으로써 (키 스트로크를 줄이기 위해) 삽입될 수 있다.

[0102] XML은 "확장 가능" 한데 왜냐하면, HTML과는 다르게, 마크업 심볼이 제한적이지 않고 자기-정의적이기 때문이다. XML은 실제적으로 어떻게 문서 구조를 만들지 표준이 되는 SGML(Standard Generalized Markup Language)의 보다 간단하고 사용하기 쉬운 서브세트이다. HTML과 XML은 많은 웹 어플리케이션에서 함께 사용될 것으로 기대된다. 예를 들어, XML 마크업은 HTML 페이지 내에서 나타날 수 있다. 이러한 점에서, 현재 본 발명에서 사용되는 특정 문법은 마크업 언어의 조합을 포함할 수 있다.

[0103] 도 4에서, 엔진(44)은 정책 표현 언어로 표현되고 디지털 포맷으로 저장된 정책(45)을 적용한다. 정책(45)은 하나 이상의 디지털 파일이나, 데이터베이스, 또는 다른 저장된 데이터 포맷으로도 나타날 수 있다. 당업자라면 디지털 파일이 어떠한 형식으로도 변환될 수 있다는 것을 인식할 것이다: 그것의 특징은 데이터베이스의 필드에 삽입될 수 있고, 또는 상기 파일은 한 포맷에서 다른 포맷으로 변환될 수도 있다. 그러므로, 적어도 초기에 정책은 일반 텍스트 파일(.txt)이나 문서(.doc) 포맷처럼 텍스트 에디터에 의해 사용될 수 있는 디지털 포맷으로 사람에 의해 최적화되어 생성될 수 있는 반면, 그러한 초기 디지털 파일은 도면부호(45)에 저장되기 전 어떤 수의 형식으로도 변환될 수 있다고 예상할 수 있다. 만일 클라이언트(40)가 요구된 인증서를 받을 자격이 있다면, 데이터 포맷에 무관하게, 그 속에 표현된 발행 정책(45)은 클라이언트(40)에 의해 만족되어야 한다. 정책(45)은 또한 생성된 인증서의 포맷을 지배할 수 있는데, 즉 자격 증명 포맷팅을 위해 정책을 구성할 수 있다.

[0104] 발행 정책(45)은 되도록 적어도 다음 내용을 구성요소로 한다.

[0105] 클라이언트 인증 요건

[0106] 클라이언트 허가 요건

[0107] 인증서 발행 서비스 허가 요건

[0108] 허가 실행 명령어

[0109] 인증서 발행 시스템의 예시를 위해 다시 한번 유명한 MICROSOFT WINDOWS® Rights Management Server 발행기에서 보면, 예를 들면 MICROSOFT® Office 2003에서의 정보 권리 관리 특징처럼, 당업자라면 상기 발행기가 보호 문서 및 이메일에 대해 "정보 권리 관리" 특징을 구현하는데 사용될 수 있음을 인정할 것이다. 솔루션의 일부로서, 현재 사용 가능한 버전의 WINDOWS® Rights Management Server는, 예상되고 하드-코드화되고 불안정한 발행 정책 정의를 포함하는 발행기를 활용한 것이다. 단지 고정된 주지의 발행 정책 세트만이 실행될 수 있다. 예를 들어:

[0110] 신뢰받는 어플리케이션은 무엇인가?

[0111] 어떤 사용자들이 특별히 제외되었는가?

[0112] 어떤 엔티티들이 사용자 신분 자격 증명을 발행하도록 신뢰되었는가?

[0113] 어떤 버전(들)의 권리 관리 소프트웨어가 사용자의 데스크탑에서 실행되어야 하는가?

[0114] 반대로, 현재 사용 가능한 버전의 WINDOWS® Rights Management Server는 새로운 발행 정책을 실행할 수는 없다, 아래와 같이:

[0115] 무엇이 사용자의 사업 분야에 있어서 신뢰받는 어플리케이션인가?

[0116] 어떤 부류의 사용자들이 특별히 제외되었는가(예를 들어, 네트워크 패스워드가 7일 이내에 만료되는 모든 사람)

[0117] 어떤 특정 자격 증명들이 인증서 발행 시스템으로부터 생성을 신뢰받는가?

[0118] 여기의 본 발명의 시스템과 방법을 구현하기 위해 현재 이용 가능한 버전의 WINDOWS® Rights Management 인증서 발행 서비스를 개조함으로써 상기 제품은 유연한 정책 표현 언어로 된 정책들을 이해하고 실행할 수 있으며, 위에 수록된 새로운 발행 정책 뿐만 아니라 임의의 다른 예상 가능한 발행 정책도 배치된 발행기를 수정하지 않고 수용될 수 있다. 단지 발행 정책(45)에 표현된 정책만 바뀌면 된다.

[0119] 인증서 발행 시스템은 발행 정책(45)을, 사용 가능한 발행된 인증서 포맷 세트와 함께 발행하여, 클라이언트(40)로 하여금 발견(discovery) 프로세스, 포렌식스(forensic) 분석 등을 수행하게 할 수 있다. 다음은 기술한 목적을 가진 전형적인 발행 정책이다.

[0120]
```
<r:license licenseId="f34e026a-836c-4557-97db-4368b3eddd14"
xmlns:r="urn:mpeg:mpeg21:2003:01-REL-R-NS">
```

[0121]
```
    <r:title>RightsAccountCertificateIssuancePolicyRoot-Public</r:title>
```

[0122]
```
    <r:inventory>
```

[0123]
```
        <r:forAll licensePartId="LP0" varName="ValidityInterval">
```

[0124]
```
            <r:anXmlExpression>/r:validityInterval</r:anXmlExpression>
```

[0125]
```
        </r:forAll>
```

[0126]
```
        <r:forAll licensePartId="LP1" varName="AccountEncryptionPublicKey" />
```

[0127]
```
        <r:keyHolder licensePartId="LP2">
```

[0128] <r:info>

[0129] <KeyName xmlns="http://www.w3.org/2000/09/xmldsig#">RACIssuancePolicyRoot</KeyName>

[0130] <KeyValue xmlns="http://www.w3.org/2000/09/xmldsig#">

[0131] <RSAKeyValue>

[0132] <Modulus>rrREhbeyebCOsKWeVh7KSc6oFJj6zZX8vJQQDKWxpDjwm7EvbSSgwt/3/ZVN5QJa8vc1Z061gkp5hRGCs1VvJzSVs+duOcaz519uQXTCXft0tVuQkv7LCktbT5aKOpUuoDs26Hs/Vw4Cg4IJwbMAmyuAZ27o6ngd1L1Vm7o/rr0=</Modulus>

[0133] <Exponent>AQAB</Exponent>

[0134] </RSAKeyValue>

[0135] </KeyValue>

[0136] </r:info>

[0137] </r:keyHolder>

[0138] <r:forAll licensePartId="LP3" varName="PrivateKey">

[0139] <r:anXmlExpression>/tm:privateKey/xkms:RSAKeyValue</r:anXmlExpression>

[0140] </r:forAll>

[0141] <r:forAll licensePartId="LP4" varName="Licensor">

[0142] <r:propertyPossessor>

[0143] <trustedLicensor xmlns="tm" />

[0144] <r:trustedRootIssuers>

[0145] <r:keyHolder licensePartIdRef="LP2" />

[0146] </r:trustedRootIssuers>

[0147] </r:propertyPossessor>

[0148] </r:forAll>

[0149] <r:forAll licensePartId="LP5" varName="SidPrincipal">

[0150] <r:anXmlExpression>/tm:sidPrincipal</r:anXmlExpression>

[0151] </r:forAll>

[0152] </r:inventory>

[0153] <r:grant>

[0154] <r:forAll licensePartIdRef="LP4" />

[0155] <r:forAll licensePartIdRef="LP1" />

[0156] <r:forAll licensePartIdRef="LP0" />

[0157] <r:forAll varName="AccountInfo">

[0158] <r:anXmlExpression>/tm:account/tm:identity[@type="urn:msft:tm:identity:rfc822" and contains("microsoft.com")]</r:anXmlExpression>

[0159] </r:forAll>

[0160] <r:principal varRef="Licensor" />

[0161] <r:issue />

[0162] <r:grant>

[0163] <r:keyHolder varRef="AccountEncryptionPublicKey" />

[0164] <r:possessProperty />

[0165] <account varRef="AccountInfo" xmlns="http://www.microsoft.com/DRM/XrML2/TM/v2" />

[0166] <r:validityInterval varRef="ValidityInterval" />

[0167] </r:grant>

[0168] <r:validityInterval>

[0169] <r:notBefore>2004-05-20T09:48:49Z</r:notBefore>

[0170] <r:notAfter>2004-05-20T09:48:49Z</r:notAfter>

[0171] </r:validityInterval>

[0172] </r:grant>

[0173] <r:grant>

[0174] <r:forAll licensePartIdRef="LP4" />

[0175] <r:forAll licensePartIdRef="LP1" />

[0176] <r:forAll licensePartIdRef="LP0" />

[0177] <r:forAll varName="KeyUsageInfo">

[0178] <r:anXmlExpression>/tm:keyUsage[@uri="urn:msft:tm:keyUsage:encryption" or @uri="urn:msft:tm:keyUsage:signing"]</r:anXmlExpression>

[0179] </r:forAll>

[0180] <r:principal varRef="Licensor" />

[0181] <r:issue />

[0182] <r:grant>

[0183] <r:keyHolder varRef="AccountEncryptionPublicKey" />

[0184] <r:possessProperty />

[0185] <keyUsage varRef="KeyUsageInfo" xmlns="http://www.microsoft.com/DRM/XrML2/TM/v2" />

[0186] <r:validityInterval varRef="ValidityInterval" />

[0187] </r:grant>

[0188] </r:grant>

[0189] <r:grant>

[0190] <r:forAll licensePartIdRef="LP4" />

[0191] <r:forAll licensePartIdRef="LP1" />

[0192] <r:forAll licensePartIdRef="LP0" />

[0193] <r:forAll varName="BindingPrincipalInfo">

[0194] <r:anXmlExpression>/tm:bindingPrincipals/r:allPrincipals/tm:sidPrincipal</r:anXmlExpression>

[0195] </r:forAll>

[0196] <r:principal varRef="Licensor" />

[0197] <r:issue />

[0198] <r:grant>

[0199] <r:keyHolder varRef="AccountEncryptionPublicKey" />

[0200] <r:possessProperty />

[0201] <bindingPrincipals varRef="BindingPrincipalInfo" xmlns="http://www.microsoft.com/DRM/XrML2/TM/v2" />

[0202] <r:validityInterval varRef="ValidityInterval" />

[0203] </r:grant>

[0204] </r:grant>

[0205] <r:grant>

[0206] <r:forAll licensePartIdRef="LP4" />

[0207] <r:forAll licensePartIdRef="LP1" />

[0208] <r:forAll licensePartIdRef="LP0" />

[0209] <r:forAll varName="CertificationPrincipalInfo">

[0210] <r:anXmlExpression>/tm:certificationPrincipals/r:allPrincipals/tm:sidPrincipal</r:anXmlExpression>

[0211] </r:forAll>

[0212] <r:principal varRef="Licensor" />

[0213] <r:issue />

[0214] <r:grant>

[0215] <r:keyHolder varRef="AccountEncryptionPublicKey" />

[0216] <r:possessProperty />

[0217] <certificationPrincipals varRef="CertificationPrincipalInfo" xmlns="http://www.microsoft.com/DRM/XrML2/TM/v2" />

[0218] <r:validityInterval varRef="ValidityInterval" />

[0219] </r:grant>

[0220] </r:grant>

[0221] <r:grant>

[0222] <r:forAll licensePartIdRef="LP4" />

[0223] <r:forAll licensePartIdRef="LP1" />

[0224] <r:forAll licensePartIdRef="LP0" />

[0225] <r:forAll varName="TrustedSecurityProcessor">

[0226] <r:propertyPossessor>

[0227] <trustedSecurityProcessor xmlns="tm" />

```
[0228]                      <r:trustedRootIssuers>
[0229]                          <r:keyHolder licensePartIdRef="LP2" />
[0230]                      </r:trustedRootIssuers>
[0231]                  </r:propertyPossessor>
[0232]              </r:forAll>
[0233]              <r:principal varRef="Licensor" />
[0234]              <r:issue />
[0235]              <r:grant>
[0236]                  <r:keyHolder varRef="TrustedSecurityProcessor" />
[0237]                  <r:possessProperty />
[0238]                  <holdsPrivateKey xmlns="http://www.microsoft.com/DRM/XrML2/TM/v2">
[0239]                      <r:keyHolder varRef="AccountEncryptionPublicKey" />
[0240]                  </holdsPrivateKey>
[0241]                  <r:validityInterval varRef="ValidityInterval" />
[0242]              </r:grant>
[0243]          </r:grant>
[0244]          <r:grant>
[0245]              <r:forAll licensePartIdRef="LP4" />
[0246]              <r:forAll licensePartIdRef="LP1" />
[0247]              <r:forAll licensePartIdRef="LP0" />
[0248]              <r:principal varRef="Licensor" />
[0249]              <r:issue />
[0250]              <r:grant>
[0251]                  <r:forAll varName="TrustedSecurityProcessor">
[0252]                      <r:propertyPossessor>
[0253]                          <trustedSecurityProcessor xmlns="tm" />
[0254]                          <r:trustedRootIssuers>
[0255]                              <r:keyHolder licensePartIdRef="LP2" />
[0256]                          </r:trustedRootIssuers>
[0257]                      </r:propertyPossessor>
[0258]                  </r:forAll>
[0259]                  <r:principal varRef="TrustedSecurityProcessor" />
[0260]                  <receivePrivateKey xmlns="tm" />
[0261]                  <r:keyHolder varRef="AccountEncryptionPublicKey" />
[0262]                  <r:validityInterval varRef="ValidityInterval" />
[0263]              </r:grant>
```

[0264] </r:grant>

[0265] <r:grant>

[0266] <r:forAll licensePartIdRef="LP4" />

[0267] <r:forAll licensePartIdRef="LP3" />

[0268] <r:forAll licensePartIdRef="LP5" />

[0269] <r:forAll licensePartIdRef="LP0" />

[0270] <r:principal varRef="Licensor" />

[0271] <r:issue />

[0272] <r:grant>

[0273] <r:allPrincipals>

[0274] <sidPrincipal varRef="SidPrincipal" xmlns="http://www.microsoft.com/DRM/XrML2/TM/v2" />

[0275] </r:allPrincipals>

[0276] <decryptWithPrivateKey xmlns="tm" />

[0277] <privateKey varRef="PrivateKey" xmlns="http://www.microsoft.com/DRM/XrML2/TM/v2" />

[0278] <r:validityInterval varRef="ValidityInterval" />

[0279] </r:grant>

[0280] </r:grant>

[0281] <r:grant>

[0282] <r:forAll licensePartIdRef="LP4" />

[0283] <r:forAll licensePartIdRef="LP3" />

[0284] <r:forAll licensePartIdRef="LP5" />

[0285] <r:forAll licensePartIdRef="LP0" />

[0286] <r:principal varRef="Licensor" />

[0287] <r:issue />

[0288] <r:grant>

[0289] <r:allPrincipals>

[0290] <sidPrincipal varRef="SidPrincipal" xmlns="http://www.microsoft.com/DRM/XrML2/TM/v2" />

[0291] </r:allPrincipals>

[0292] <signWithPrivateKey xmlns="tm" />

[0293] <privateKey varRef="PrivateKey" xmlns="http://www.microsoft.com/DRM/XrML2/TM/v2" />

[0294] <r:validityInterval varRef="ValidityInterval" />

[0295] </r:grant>

[0296] </r:grant>

[0297] <r:issuer xmlns:r="urn:mpeg:mpeg21:2003:01-REL-R-NS">

[0298] <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">

[0299] <SignedInfo>

[0300] <CanonicalizationMethod Algorithm="http://www.microsoft.com/xrml/lwc14n" />

[0301] <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />

[0302] <Reference>

[0303] <Transforms>

[0304] <Transform Algorithm="urn:mpeg:mpeg21:2003:01-REL-R-NS:licenseTransform" />

[0305] <Transform Algorithm="http://www.microsoft.com/xrml/lwc14n" />

[0306] </Transforms>

[0307] <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />

[0308] <DigestValue>ZYG/bsCMxs5FhAT/atoXYGRuQ6Y=</DigestValue>

[0309] </Reference>

[0310] </SignedInfo>

[0311] <SignatureValue>Jd72Kt3h1fTYigxaYlrjaES+RLzmMZjr7bJBb9236GD7tty90zmZQxpYqTrA9D/qmrca5k84BGzefXodP8uLokDxUkpdXEI4aVurCDjP7cHwtOnZdMR5ATMvSgPn4kLHZ6E0g2pX7gjAm8jItvmD49Sa2D9CKjOtORq5zEkQMLc=</SignatureValue>

[0312] <KeyInfo>

[0313] <KeyValue>

[0314] <RSAKeyValue>

[0315] <Modulus>rrREhbeyebCOsKWeVh7KSc6oFJj6zZX8vJQQDKWxpDjwm7EvbSSgwt/3/ZVN5QJa8vclZ061gkp5hRGCs1VvJzSVs+duOcaz519uQXTCXft0tVuQkv7LCktbT5aKOpUuoDs26Hs/Vw4Cg4IJwbMAmyuAZ27o6ngd1L1Vm7o/rr0=</Modulus>

[0316] <Exponent>AQAB</Exponent>

[0317] </RSAKeyValue>

[0318] </KeyValue>

[0319] </KeyInfo>

[0320] </Signature>

[0321] <r:details>

[0322] <r:timeOfIssue>2004-05-20T09:48:49Z</r:timeOfIssue>

[0323] </r:details>

[0324] </r:issuer>

[0325] <r:otherInfo>

[0326] <tm:infoTables xmlns:tm="http://www.microsoft.com/DRM/XrML2/TM/v2">

[0327] <tm:infoList tag="#LicenseType">

[0328] <tm:infoStr name="licenseRole">RightsAccountCertificateIssuancePolicy</tm:infoStr>

[0329] <tm:infoStr name="licenseVersion">1.0</tm:infoStr>

[0330] <tm:infoStr

name="licenseType">RightsAccountCertificateIssuancePolicy-Public</tm:infoStr>

[0331] <tm:infoStr name="licensorUrl">http://rms.microsoft.com/rms/certification</tm:infoStr>

[0332] </tm:infoList>

[0333] </tm:infoTables>

[0334] </r:otherInfo>

[0335] </r:license>

[0336] 도 5는 정책 표현 언어 및/또는 정책 언어 파싱 및 실행 엔진(56)에 의해 지원되지 않는 새로운 발행 정책들을 적용하고 실행하기 위해, 발행 컴포넌트를 확장하는 시스템과 방법을 나타낸다. 예를 들어 부가 데이터(59)가 발행 정책 데이터(58)와 결합함으로써 정책 표현 언어 문법이 확장될 때, 확장된 정책 표현 언어 구조의 의미를 다루는 플러그인 논리(55)가 실행 엔진(56)에 부가될 수 있다. 본래의 정책 언어 와일드카드가 사용될 때, 예를 들어 와일드카드 대체 패턴 데이터(60)를 발행 정책 데이터(58)에 부가함으로써, 확장하는 와일드카드 대체 패턴(60)을 다루는 플러그인 논리(57)가 실행 엔진(56)에 부가될 수 있다.

[0337] 데이터-구동의 정책 평가 엔진(56)은 이해되지 않거나 처리되도록 의도하지 않은 인증서의 의미를 예상하도록 설계될 수 있다. 이러한 인증서 의미에 대한 확장성 메커니즘이 존재하여 그러한 인증서에 대한 플러그인 논리(55 또는 57) 등의 커스텀 논리가 호출되어 값을 제공하여, 커스텀 프로세싱을 수행하거나, 아니면 알려지지 않은 의미를 잘 정의된 방법으로 다루게 된다. 이러한 처리의 결과는 정책 평가 엔진(56)에 피드백되어 인증서를 받을 자격이 있는가에 관한 결정의 최종 결과에 가능한 한 영향을 미치게 된다.

[0338] 여기에서 발행 컴포넌트를 확장하도록 고려된 적어도 두 개의 메커니즘이 있다. 첫째, 확장할 수 있는 정책 표현 언어(예를 들어, XML의 확장성을 강화하는 것)를 세우고 플러그인 논리(55)와 같은 적절한 플러그인 메커니즘을 제공함으로써, 인증서 발행 시스템은 고객에게 커스텀 확장성을 지원할 수 있다. 둘째, 정책 표현 언어에 와일드카드 대체 패턴의 개념을 포함시키고 플러그인 논리(57)와 같은 적절한 플러그인 메커니즘을 제공함으로써, 발행기가 고객에게 커스텀 확장성을 지원할 수 있다.

[0339] 발행 컴포넌트를 확장하기 위한 전술한 옵션들 중 첫째 것으로 시작할 때, 정책 표현 언어 문법은 바람직한 실시예에서 확장성이 있다. 정책 실행 엔진(56)은 정책 표현 언어의 원래 양상의 어의 의미를 어떻게 취급하는지 알도록 사전 구성될 수 있다. 그러나, 정책 표현 언어의 의미가 확장된다면, 상기 엔진(56) 역시 확장될 수 있다.

[0340] 정책 표현 언어의 확장은 엔진(56)에 의해 액세스 가능한 디지털 데이터(59)에 설명될 수 있다. 그러한 확장(59)은 하나 또는 그 이상의 파일이나 데이터베이스 또는 어떠한 다른 저장된 데이터 포맷으로 명시될 수 있다. 확장은 일반 텍스트 파일(.txt)이나 문서(.doc) 포맷처럼 텍스트 에디터에 의해 사용될 수 있는 포맷으로 사람에 의해 최적화되어 생성될 수 있는 반면, 그러한 초기 파일은 저장되기 전 임의의 수의 형식으로도 변환될 수 있다. 데이터 포맷에 무관하게, 그곳에 표현된 발행 정책 확장(59)은 확장된 정책 표현 문법(59)을 사용하여 표현된 정책들을 적용하고 실행하기 위해서, 커스텀 논리(55)에 의해 액세스 가능하다.

[0341] 이것을 이루기 위해서, 엔진(56)은 플러그인 논리(55)와 같은 확장된 논리("플러그인"으로 알려진)가 등록되는 것을 인정하도록 구성될 수 있다. 플러그인(55)은 정책 표현 언어의 어느 새로운 문법적 확장(59)을 의미적으로 보조하기 위해 제공될 수 있다. 그러므로 예상되지 않은 고객 요구들에 대해 엔진(56) 그 자체나 정책 표현 언어에서 원래 사용되던 문법을 철저히 조사할 필요 없이도 응답이 가능하다. 정책 표현 언어는 문장 구성을 통해 확장을 지원할 수 있게 이상적으로 설계된다.

[0342] 하나의 예시가 플러그인(55)을 이용한 엔진(55)의 확장성을 명확히 하는데 도움을 줄 수 있다. 인증서 발행 시스템은 사용자의 PKI 아이덴티티를 나타내는 요소를 포함하는 XML 정책 표현 언어를 가진다고 가정하라. 아마도 다음과 같을 것이다.

[0343] <user>

[0344] <name>George Washington</name>

[0345] <publickey>1234567890</publickey>

[0346] </user>

[0347] 인증서 발행 시스템 고객은 사용자의 개념을 확장하여 사용자의 엔터프라이즈 LDAP(Lightweight Directory Access Protocol) 아이덴티티를 포함하고자 할 수 있다. LDAP 문법을 포함하기 위한 정책 언어 확장은 필요한 경우 인증서 발행 시스템 데이터베이스에서 LDAP 질의를 수행하는 코드에 의해 지원될 수 있다. 이 경우, 확장된 정책 언어 구조는 다음과 같을 것이다.

[0348] <user>

[0349] <name>George Washington</name>

[0350] <publickey>1234567890</publickey>

[0351] <extension:ldapid>georgewashington</extension:ldapid>

[0352] </user>

[0353] 부가적으로, LDAP 질의로 사용자를 증명한 코드, 즉 플러그인(55)은 컴파일되어 인증서 발행 시스템에 등록될 것이다. 상기 플러그인(55)은 확장된 정책 표현 언어 문법을 만났을 때 엔진(56)에 의해 불리게 될 것이다. 일부 실시예에서, 플러그인을 등록하지 않고 확장된 문법을 사용하는 것이 가능할 수도 있다는 것을 유의하라. 이는 동일한 확장 문법을 입력/출력 인증서와 엔진에 의해 해석되는 발행 정책 데이터 모두에 부가함으로써 이루어질 수 있다.

[0354] 발행 컴포넌트를 확장하는 두 번째 옵션으로 돌아오면, 사용되는 정책 표현 언어는 와일드카드 대체 파라미터를 포함할 수 있다. 와일드카드 대체 패턴은 본래의 발행 정책(58)으로 설명될 수 있고, 또는 본래의 발행 정책(58)을 보충하기 위하여 부가적 데이터(60)를 이용 가능하게 함으로써 정책에 부가될 수도 있다.

[0355] 와일드카드 대체 패턴(60)은 하나 또는 그 이상의 파일들, 데이터베이스 또는 어느 다른 저장된 데이터 포맷으로 명시될 수 있다. 초기 대체 패턴은 일반 텍스트 파일(.txt)이나 문서(.doc) 포맷처럼 텍스트 에디터에 의해 사용될 수 있는 포맷으로 사람에 의해 최적화되어 생성되는 반면, 그러한 초기 파일은 대체 패턴(60)에 저장되기 전 어떤 수의 형식으로도 변환될 수 있다. 데이터의 포맷에 무관하게, 그곳에 표현된 와일드카드 대체 패턴(60)은 커스텀 논리(57)에 의해 지시되는 방식으로 와일드카드를 적용하고 실행하기 위해, 커스텀 논리(57)에 의해 액세스 가능하다.

[0356] 만일 정책 표현 언어가 그 문법 내에 와일드카드 정의(60)를 포함하고 있고 엔진(56)이 특정의 원하는 값을 선택하기 위해 커스텀 논리(57)를 등록하는 매커니즘을 제공한다면, 이것은 인증서 발행 시스템의 확장성을 구현하는 또 하나의 방법을 제공하는 것이다.

[0357] 나아가서, 예시 하나가 이를 명료화하는데 도움이 될 수 있다. 전형적인 인증서 발행 시스템은 발행된 인증서의 포맷을 정의하는 정책을 포함할 수 있다. 예를 들어 서비스는, 상기 서비스가 "신뢰받는 피고용인 인증서(trusted employee certificates)"를 클라이언트에게 발행할 수 있다고 언급하는 인증서 발행 정책을 가질 수 있다. 발행기는 클라이언트의 동적 활동에 대해 응답해야만 하기 때문에, 상기 와일드카드 인증서 발행 정책은 다음과 같은 구조를 가질 것이다:

[0358] 인증서 발행 시스템은 "신뢰받는 피고용인 인증서"를 적절하다고 생각되는 임의의 클라이언트에게 발행할 수 있다.

[0359] 인증서 발행 시스템 소유자는 그 다음 "적절하다고 생각되는" 부분에 어떤 특정 클라이언트를 채우는 논리, 예를 들어 플러그인 논리(57)를 정의하고 등록할 수 있다. 상기 논리(57)는 특정 서비스의 실시 동안에 어떤 클라이언트에게 "신뢰받는 피고용인 인증서"를 발행해야 할지 결정할 수 있다. 이 논리(57)는 인증서 발행 정책

에 있어서 더 일반적인 와일드카드 정의 조항에 마주쳤을 때 인증서 발행 시스템에 의해 호출될 수 있다.

[0360] 도 1과 2에 의해 제공되는 일반적인 구조에 따라 구성되는 다양한 연산 환경들을 고려할 때, 여기에서 제시하는 시스템과 방법들은 어떤 방법으로든 특정의 연산 아키텍쳐로 한정되는 것으로 추론될 수 없다. 대신에, 본 발명은 어떤 단일한 실시예에 한정되는 것이 아니며, 부가된 청구항과 관련한 폭과 범위로 추론되어야 한다.

발명의 효과

[0361] 본 발명에 의하면 유입되는 인증서를 일반 포맷으로 변환하고, 출력되는 생성 인증서를 임의의 지원된 포맷으로 변환하기 위한 변환 컴포넌트를 포함하는 개선된 인증서 발행 시스템을 제공할 수 있다.

도면의 간단한 설명

[0001] 도 1은 본 발명의 다양한 형태와 관련하여 사용하기에 적합한 전형적인 연산 장치의 기본 특징들을 개략적으로 나타내는 블록도 이다. 상기 연산 장치는 컴퓨터 판독가능한 매체의 명령어에 액세스하여, 상기 명령어를 인증서 발행 시스템의 기능을 수행하기 위한 적절한 순서로 실행하게 된다.

[0002] 도 2는 인증서 발행 시스템이 동작하는 전형적인 네트워크 연산 환경을 나타낸다. 발행기는 예를 들어 장치(271)에 존재할 수 있다. 장치(277)에 있는 클라이언트 프로세스는(271)에 있는 발행기로부터 인증서를 요청할 수 있고, 그 후에는 장치(275)에 있는 서버 프로세스와 통신하여 상기 인증서를 사용할 수 있다.

[0003] 도 3a는 인증서 발행 시스템의 변환 컴포넌트 부분을 나타낸다. 상기 변환 구성요소, 즉 변환 엔진은, 들어오는 복수 포맷의 인증서들을 단일의 일반 포맷으로 변환시키는 역할을 수행한다. 발행 컴포넌트(34)에 의해서 클라이언트에게 전달을 위한 일반 포맷으로 생성된 인증서들은 필요한 경우 어떤 포맷으로도 변환될 수 있다.

[0004] 도 3b는 들어오는 인증서 요구가 발행기에 의해 어떻게 처리되는지에 관한 하나의 형태를 나타낸다. 우선, 상기 요구가 서버 엔트리 포인트(301)를 히트한다. 자신의 독자적인 포맷으로 되어있는 상기 인증서 또는 다른 요구 정보, 예를 들어 제 1 인증서 포맷(302)은, 인증서 변환 층(303)에 전달되어 제 2 포맷(304)으로 변환된다. 그 다음, 상기 인증서 및/또는 요구(304)와 연관된 다른 정보는 서버 인증서 엔진(305)에 의해 다루어진다.

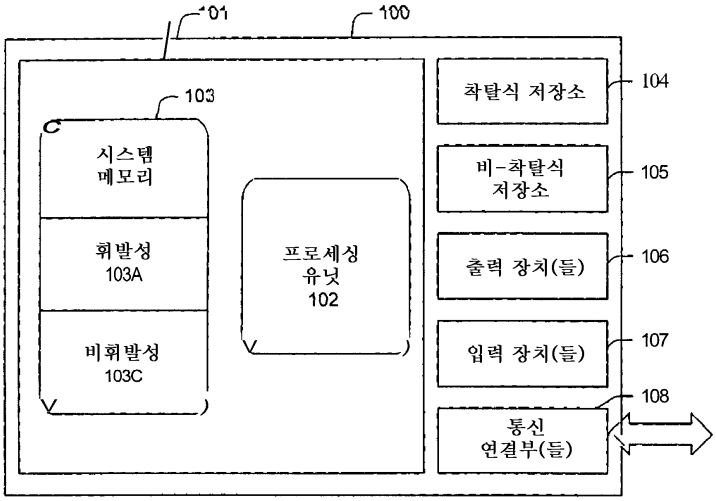
[0005] 도 3c는 클라이언트(310)가 서버(313)에게 인증서 요구를 보내는 본 발명의 다른 형태를 나타낸다. 타입 1 인증서는 클라이언트(310)의 요구와 연관되어 있다. 상기 타입 1 인증서는 인증서 변환 엔진(CTE)(311)에 의해 타입 2 인증서로 변환될 수 있다. 상기 타입 2 인증서는 발행 엔진(312)에 의해 사용되어 클라이언트(310)의 요구를 승낙할지 결정할 수 있다. 상기 발행 엔진(312)가 제 2 인증서 포맷으로 인증서를 발행한 때, 상기 CTE(311)는 그것을 클라이언트(310)를 위한 제 1 포맷으로 변환할 수 있다.

[0006] 도 4는 인증서 발행 시스템의 발행 컴포넌트 부분을 나타낸다. 인증(41), 허가(42), 및 자격 증명 포맷팅(43)은 상기 발행 컴포넌트에 의해 수행되는 전형적인 기능들이다. 이러한 기능들을 수행하는 것 중 한 부분으로, 범용 정책 언어 파싱 및 실행 엔진(44)은 인증서 발행 정책(45)을 적용할 수 있다. 그러므로, 실행될 정책이 엔진(44)에서 표시될 필요가 없고, 실행시에 엔진(44)에 의해 소비되는 발행 정책(45) 내에 표시되면 된다는 점에서 볼 때에, 상기 발행 컴포넌트는 데이터-구동된다고 할 수 있다.

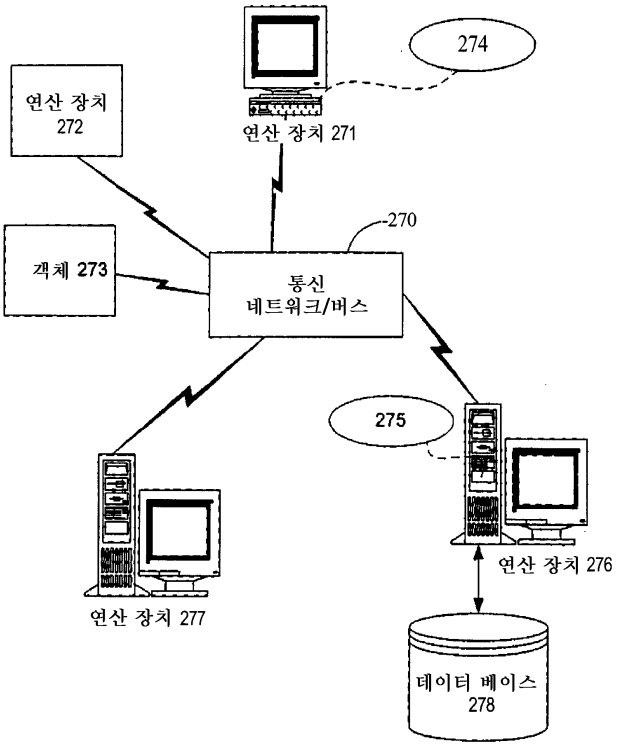
[0007] 도 5는 발행 컴포넌트로 하여금 정책 표현 언어 및/또는 정책 언어 파싱 및 실행 엔진(56)에 의해 원래부터 지원되지 않는 새로운 발행 정책을 적용하고 실행하기 위해 상기 발행 컴포넌트를 연장하는 시스템 및 방법들을 나타낸다. 예를 들어 발행 정책 메타 정보(59)에서 정책 언어 문법이 연장될 때, 연장된 정책 언어 구조의 의미를 처리하는 플러그인 논리가 실행 엔진(56)에 부가될 수 있다. 원래의 정책 언어 와일드카드가 사용될 때, 확장하는 와일드 카드 대체 패턴을 처리하는 플러그인 논리(57)가 실행 엔진(56)에 부가될 수 있다.

도면

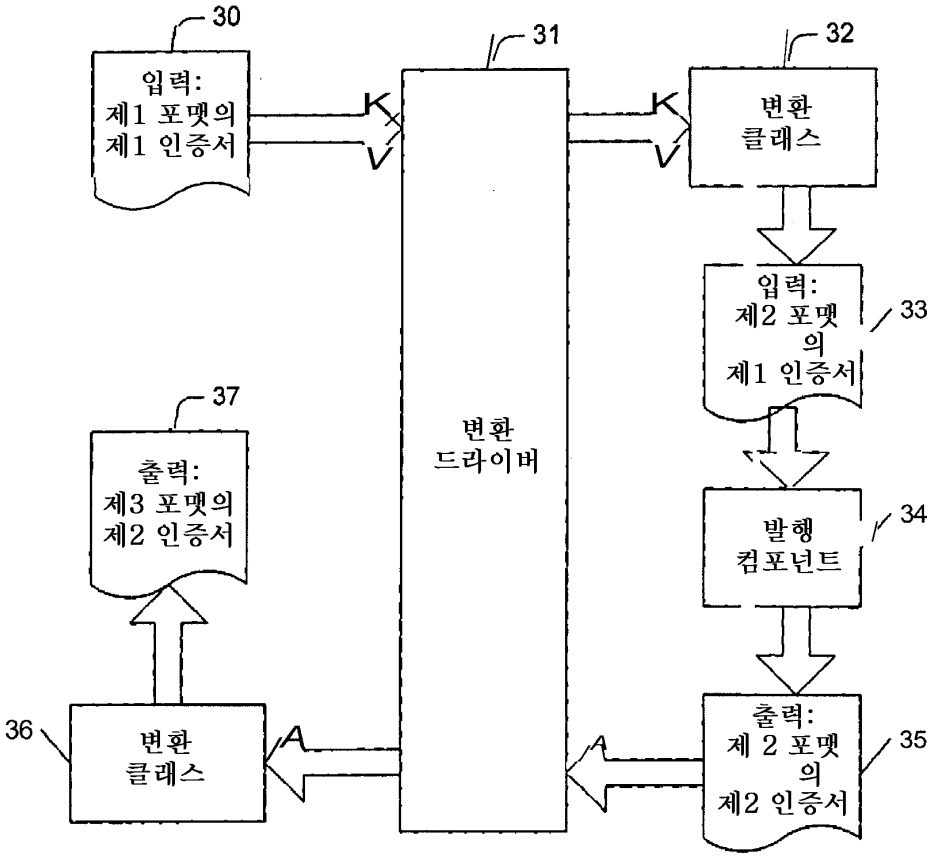
도면1


101
100
103
시스템
메모리
휘발성
103A
비휘발성
103C
프로세싱
유닛
102
착탈식 저장소
104
비-착탈식
저장소
105
출력 장치(들)
106
입력 장치(들)
107
통신
연결부(들)
108

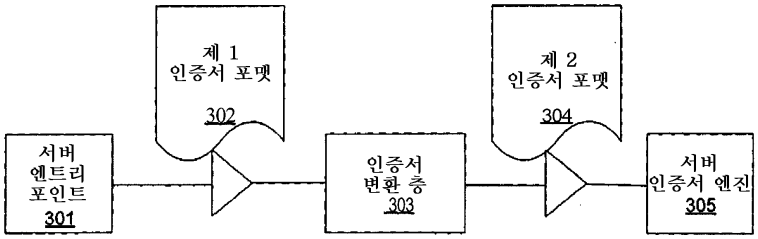
도면2


274
연산 장치
272
연산 장치 271
270
객체 273
통신
네트워크/버스
275
연산 장치 276
연산 장치 277
데이터 베이스
278

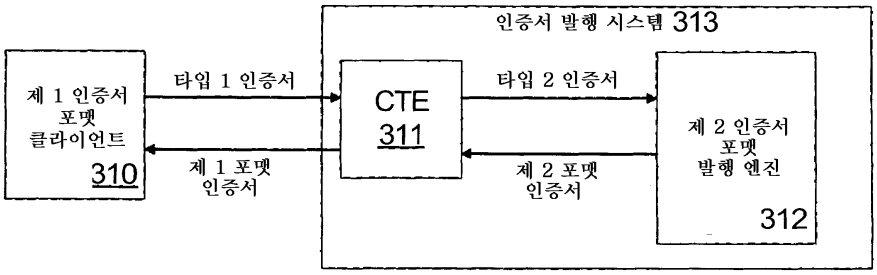
도면3a


30
입력:
제1 포맷의
제1 인증서
31
변환
드라이버
32
변환
클래스
K
V
K
V
입력:
제2 포맷
의
제1 인증서
33
발행
컴포넌트
34
출력:
제 2 포맷
의
제2 인증서
35
A
A
36
변환
클래스
37
출력:
제3 포맷의
제2 인증서

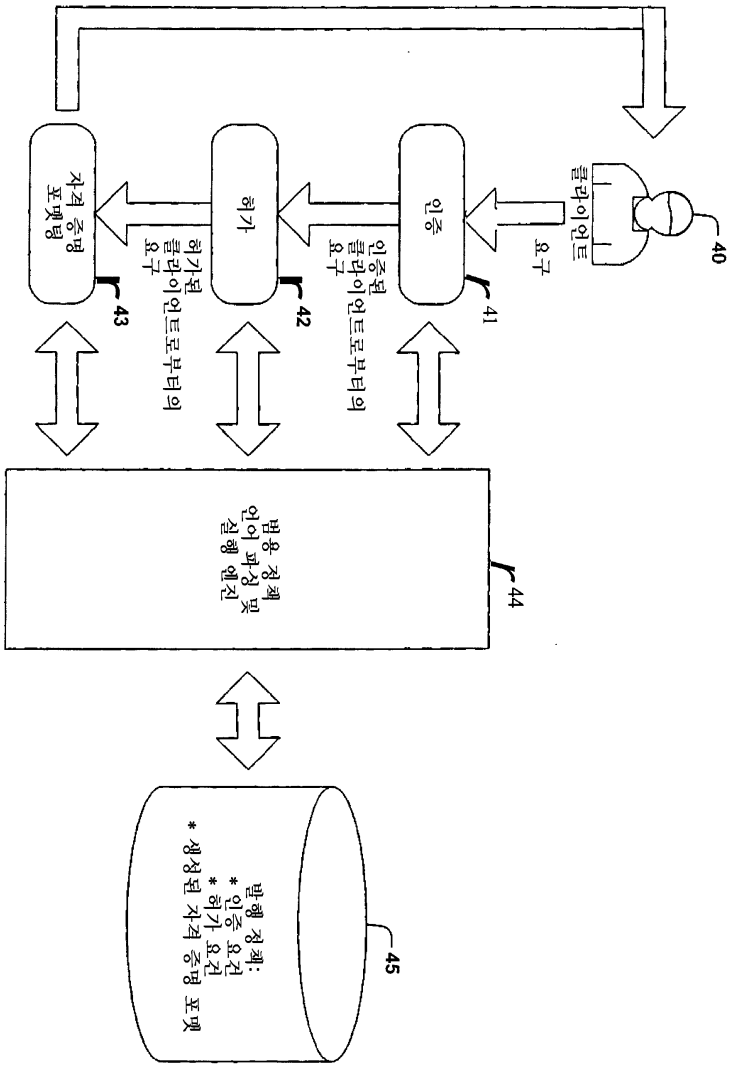
도면3b


서버
엔트리
포인트
301
제 1
인증서 포맷
302
인증서
변환 층
303
제 2
인증서 포맷
304
서버
인증서 엔진
305

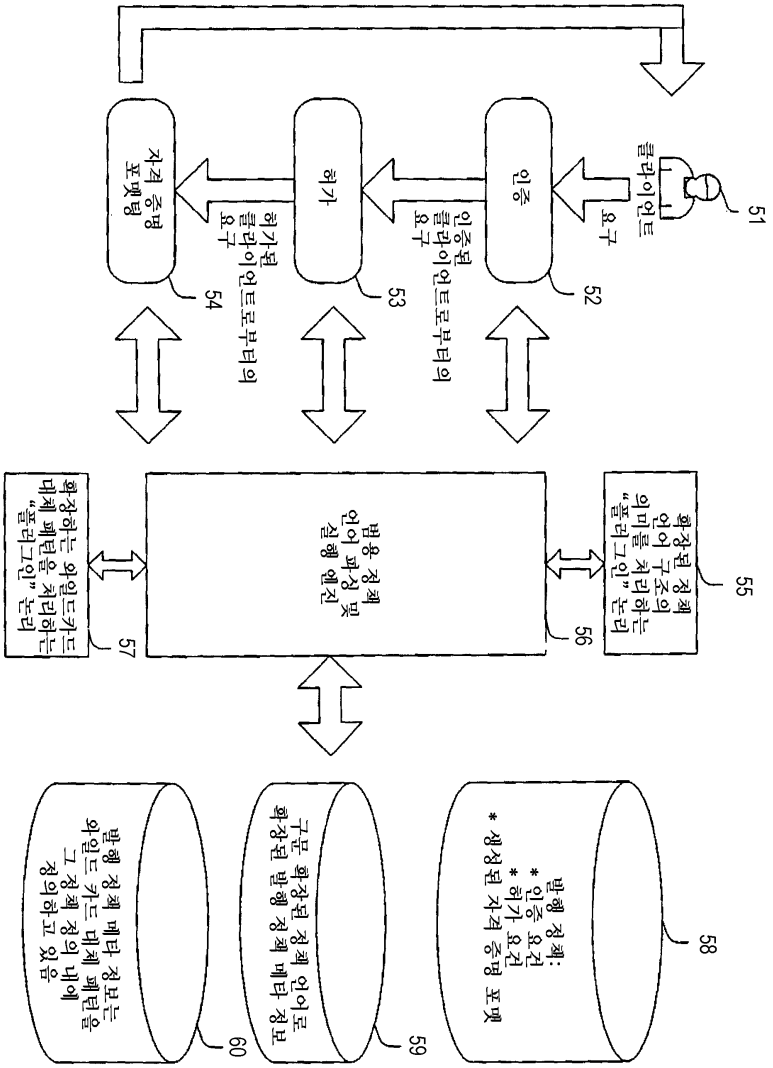
도면3c


인증서 발행 시스템 313
제 1 인증서
포맷
클라이언트
310
타입 1 인증서
제 1 포맷
인증서
CTE
311
타입 2 인증서
제 2 포맷
인증서
제 2 인증서
포맷
발행 엔진
312

도면4


클라이언트
40
요구
인증
41
인증된
클라이언트로부터의
요구
허가
42
허가된
클라이언트로부터의
요구
자격 증명
포맷팅
43
범용 정책
언어 파싱 및
실행 엔진
44
발행 정책:
* 인증 요건
* 허가 요건
* 생성된 자격 증명 포맷
45

도면5


51
클라이언트
요구
인증
52
인증된
클라이언트로부터의
요구
허가
53
허가된
클라이언트로부터의
요구
자격 증명
포맷팅
54
확장된 정책
언어 구조의
의미를 처리하는
"플러그인" 논리
55
범용 정책
언어 파싱 및
실행 엔진
56
확장하는 와일드카드
대체 패턴을 처리하는
"플러그인" 논리
57
발행 정책:
* 인증 요건
* 허가 요건
* 생성된 자격 증명 포맷
58
구문 확장된 정책 언어로
확장된 발행 정책 메타 정보
59
발행 정책 메타 정보는
와일드 카드 대체 패턴을
그 정책 정의 내에
정의하고 있음
60