(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2021년02월01일
(11) 등록번호 10-2210429
(24) 등록일자 2021년01월26일

(51) 국제특허분류(Int. Cl.)
H04L 29/06 (2006.01)
(52) CPC특허분류
H04L 63/0884 (2013.01)
H04L 63/10 (2013.01)
(21) 출원번호 10-2019-0050894
(22) 출원일자 2019년04월30일
심사청구일자 2019년04월30일
(65) 공개번호 10-2020-0126794
(43) 공개일자 2020년11월09일
(56) 선행기술조사문헌
CN105554004 B*
KR1020190041784 A*
*는 심사관에 의하여 인용된 문헌

(73) 특허권자
숭실대학교산학협력단
서울특별시 동작구 상도로 369 (상도동)
(72) 발명자
김영종
인천광역시 중구 운중로14번길 28 (운남동)
김명호
서울특별시 동작구 상도로 407, 102동 904호 (상도동, 삼호아파트)
(뒷면에 계속)
(74) 대리인
특허법인 무한

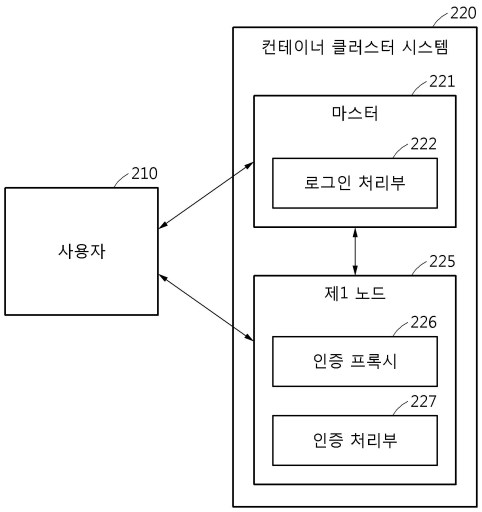
전체 청구항 수 : 총 5 항 심사관 : 문형섭

(54) 발명의 명칭 **블록체인 기반의 인증을 위한 컨테이너 클러스터 시스템**

(57) 요 약

블록체인 기반의 인증을 위한 컨테이너 클러스터 시스템이 개시된다. 일 실시예에 따르면, 마스터는 사용자로부터 로그인 정보를 포함하는 로그인 요청을 수신하고, 로그인 요청에 응답하여 사용자의 인증을 위한 인증 토큰을 사용자에게 제공하는 로그인 처리부를 포함하고, 노드들 중 사용자에게 할당된 팟을 보유한 제1 노드는 사용자로부터 인증 토큰을 수신하는 인증 프록시; 및 인증 프록시로부터 인증 토큰을 전달받아 사용자에 관한 인증을 처리하고, 인증 프록시를 통해 사용자에게 인증 처리 결과를 전송하는 인증 처리부를 포함한다.

대 표 도 - 도2



(52) CPC특허분류
H04L 67/28 (2013.01)
(72) 발명자
장성일
경기도 광주시 고불로 180-10, 105동 101호 (태전동, 디베리아빌)
조한주
서울특별시 용산구 효창원로 17, 105동 104호 (산천동, 리버힐삼성아파트)

이 발명을 지원한 국가연구개발사업
과제고유번호 2018-0-00209-001
부처명 과학기술정보통신부
과제관리(전문)기관명 정보통신기획평가원
연구사업명 SW중심대학 지원사업
연구과제명 SW중심대학(숭실대학교)
기 여 율 1/2
과제수행기관명 숭실대학교 산학협력단
연구기간 2019.01.01 ~ 2019.12.31
이 발명을 지원한 국가연구개발사업
과제고유번호 1711075727
부처명 과학기술정보통신부
과제관리(전문)기관명 정보통신기획평가원
연구사업명 대학ICT연구센터지원사업
연구과제명 초고속영상 기반의 지능형 디지털콘텐츠 플랫폼 기술연구 및 인력양성(IITP-2018-0-01419)
기 여 율 1/2
과제수행기관명 숭실대학교 산학협력단
연구기간 2018.06.01 ~ 2021.12.31

명 세 서

청구범위

청구항 1

마스터 및 노드들을 포함하는, 쿠버네티스 기반의 컨테이너 클러스터 시스템에 있어서,

상기 마스터는

사용자로부터 로그인 정보를 포함하는 로그인 요청을 수신하고, 상기 로그인 요청에 응답하여 상기 사용자의 인증을 위한 인증 토큰을 상기 사용자에게 제공하는 로그인 처리부

를 포함하고,

상기 노드들 중 상기 사용자에게 할당된 팟을 보유한 제1 노드는

상기 사용자로부터 상기 인증 토큰을 수신하는 인증 프록시; 및

상기 인증 프록시로부터 상기 인증 토큰을 전달받고, 각 사용자의 인증 정보 및 인증 기록을 저장하는 블록체인 기반의 분산 원장을 이용하여 상기 인증 토큰에 기초한 상기 사용자에 관한 인증을 처리하고, 상기 인증 프록시를 통해 상기 사용자에게 인증 처리 결과를 전송하는 인증 처리부

를 포함하고,

상기 분산 원장은 상기 마스터 및 상기 노드들 각각에 동기화되어 저장되며,

상기 블록체인은 상기 인증 프록시를 포함하는 제한된 대상에게만 액세스를 허용하는 프라이빗 블록체인인,

컨테이너 클러스터 시스템.

청구항 2

삭제

청구항 3

삭제

청구항 4

삭제

청구항 5

제1항에 있어서,

상기 프라이빗 블록체인은

비잔틴 장애를 허용하고, 적어도 4개의 구성 노드들에 의한 합의에 기초하여 블록 내 데이터를 확정하는 PBFT(practical byzantine fault tolerance) 방식에 기반하는,

컨테이너 클러스터 시스템.

청구항 6

제1항에 있어서,

상기 인증 프록시 및 상기 인증 처리부에 의해 상기 마스터의 인증 처리가 분산화되는,

컨테이너 클러스터 시스템.

청구항 7

제1항에 있어서,

상기 사용자의 서비스 가입에 대응하여 상기 사용자에게 상기 인증 프록시의 네트워크 주소가 제공되고,

상기 사용자는 상기 인증 프록시의 상기 네트워크 주소를 통해 상기 인증 프록시에 액세스하는,

컨테이너 클러스터 시스템.

청구항 8

마스터 및 노드들을 포함하는, 쿠버네티스 기반의 컨테이너 클러스터 시스템의 인증 방법에 있어서,

상기 마스터의 로그인 처리부에 의해 수행되는, 사용자로부터 로그인 정보를 포함하는 로그인 요청을 수신하고, 상기 사용자의 인증을 위한 인증 토큰을 상기 사용자에게 제공하는 단계;

상기 노드들 중 상기 사용자에게 할당된 팟을 보유한 제1 노드의 인증 프록시에 의해 수행되는, 상기 사용자로부터 상기 인증 토큰을 수신하는 단계; 및

상기 제1 노드의 인증 처리부에 의해 수행되는, 상기 인증 프록시로부터 상기 인증 토큰을 전달받고, 각 사용자의 인증 정보 및 인증 기록을 저장하는 블록체인 기반의 분산 원장을 이용하여 상기 인증 토큰에 기초한 상기 사용자에 관한 인증을 처리하고, 상기 인증 프록시를 통해 상기 사용자에게 인증 처리 결과를 전송하는 단계

를 포함하고,

상기 분산 원장은 상기 마스터 및 상기 노드들 각각에 동기화되어 저장되며,

상기 블록체인은 상기 인증 프록시를 포함하는 제한된 대상에게만 액세스를 허용하는 프라이빗 블록체인인,

컨테이너 클러스터 시스템의 인증 방법.

청구항 9

삭제

청구항 10

삭제

발명의 설명

기 술 분 야

[0001] 아래 실시예들은 블록체인 기반의 인증을 위한 컨테이너 클러스터 시스템에 관한 것이다.

[0002] 본 발명은 과학기술정보통신부 및 정보통신기획평가원의 SW중심대학사업의 연구결과로 개발된 결과물이다(2018-0-00209-001).

[0003] 본 발명은 과학기술정보통신부 및 정보통신기획평가원의 대학ICT연구센터지원사업의 연구결과로 개발된 결과물이다(IITP-2018-2018-0-01419).

배 경 기 술

[0005] 기존 컨테이너 클러스터의 구조는 마스터와 노드를 가지고 있는 분산 구조이다. 개발자나 운영자는 개발 및 관리를 위해 각 마스터 혹은 노드에 대해 접근 및 수정을 해야 하는데, 이에 대한 권한 허가 및 인증은 마스터의 API(application programming interface) 서버에서 이뤄진다. 기존 API 서버 인증 방식은 중앙 집중형 구조로써, API 서버 인증 요청이 수용량보다 증가하면 해결할 수 있는 일반적인 방법은 API 서버의 성능, 즉 마스터의

컴퓨팅 성능을 증대시키는 것이다. 하지만 이 방법은 서비스를 일시적으로 중단해야할 뿐만 아니라, 서버 성능 증대에 하드웨어적인 한계가 존재할 수 있다. 또한 인증과 관련된 기능이 API 서버에 종속적이기 때문에, 인증 요청이 많아져 인증에 지연이 발생되면 전체 컨테이너 클러스터에 영향을 줄 수 있다.

발명의 내용

해결하려는 과제

과제의 해결 수단

[0007] 일 실시예에 따르면, 마스터 및 노드들로 구성된 컨테이너 클러스터 시스템에 있어서, 상기 마스터는 사용자로부터 로그인 정보를 포함하는 로그인 요청을 수신하고, 상기 로그인 요청에 응답하여 상기 사용자의 인증을 위한 인증 토큰을 상기 사용자에게 제공하는 로그인 처리부를 포함하고, 상기 노드들 중 상기 사용자에게 할당된 팟을 보유한 제1 노드는 상기 사용자로부터 상기 인증 토큰을 수신하는 인증 프록시; 및 상기 인증 프록시로부터 상기 인증 토큰을 전달받아 상기 사용자에 관한 인증을 처리하고, 상기 인증 프록시를 통해 상기 사용자에게 인증 처리 결과를 전송하는 인증 처리부를 포함한다.

[0008] 상기 인증 처리부는 각 사용자의 인증 정보 및 인증 기록을 저장하는 블록체인 기반의 분산 원장을 이용하여 상기 인증 토큰에 기초한 상기 사용자에 관한 인증을 처리할 수 있다. 상기 분산 원장은 상기 마스터 및 상기 노드들 각각에 동기화되어 저장될 수 있다. 상기 블록체인은 상기 인증 프록시를 포함하는 제한된 대상에게만 액세스를 허용하는 프라이빗 블록체인일 수 있다. 상기 프라이빗 블록체인은 비잔틴 장애를 허용하고, 적어도 4개의 구성 노드들에 의한 합의에 기초하여 블록 내 데이터를 확정하는 PBFT(practical byzantine fault tolerance) 방식에 기반할 수 있다.

[0009] 상기 사용자의 서비스 가입에 대응하여 상기 사용자에게 상기 인증 프록시의 네트워크 주소가 제공될 수 있고, 상기 사용자는 상기 인증 프록시의 상기 네트워크 주소를 통해 상기 인증 프록시에 액세스할 수 있다. 상기 인증 프록시 및 상기 인증 처리부에 의해 상기 마스터의 인증 처리가 분산화될 수 있다.

[0010] 일 실시예에 따르면, 마스터 및 노드들로 구성된 컨테이너 클러스터 시스템의 인증 방법은, 상기 마스터의 로그인 처리부에 의해 수행되는, 사용자로부터 로그인 정보를 포함하는 로그인 요청을 수신하고, 상기 사용자의 인증을 위한 인증 토큰을 상기 사용자에게 제공하는 단계; 상기 노드들 중 상기 사용자에게 할당된 팟을 보유한 제1 노드의 인증 프록시에 의해 수행되는, 상기 사용자로부터 상기 인증 토큰을 수신하는 단계; 및 상기 제1 노드의 인증 처리부에 의해 수행되는, 상기 인증 프록시로부터 상기 인증 토큰을 전달받아 상기 사용자에 관한 인증을 처리하고, 상기 인증 프록시를 통해 상기 사용자에게 인증 처리 결과를 전송하는 단계를 포함한다.

도면의 간단한 설명

[0012] 도 1은 일 실시예에 따른 컨테이너 클러스터 시스템의 구조를 나타낸 도면.

도 2는 일 실시예에 따른 컨테이너 클러스터 시스템의 인증 구조를 나타낸 도면.

도 3은 일 실시예에 따른 중앙 집중형 인증 방식의 컨테이너 클러스터 시스템을 나타낸 도면.

도 4는 일 실시예에 따른 분산형 인증 방식의 컨테이너 클러스터 시스템을 나타낸 도면.

도 5는 일 실시예에 따른 컨테이너 클러스터 시스템의 인증 방법을 나타낸 도면.

발명을 실시하기 위한 구체적인 내용

[0013] 본 명세서에 개시되어 있는 본 발명의 개념에 따른 실시예들에 대해서 특정한 구조적 또는 기능적 설명들은 단지 본 발명의 개념에 따른 실시예들을 설명하기 위한 목적으로 예시된 것으로서, 본 발명의 개념에 따른 실시예들은 다양한 형태로 실시될 수 있으며 본 명세서에 설명된 실시예들에 한정되지 않는다.

[0014] 본 발명의 개념에 따른 실시예들은 다양한 변경들을 가할 수 있고 여러 가지 형태들을 가질 수 있으므로 실시예들을 도면에 예시하고 본 명세서에 상세하게 설명하고자 한다. 그러나, 이는 본 발명의 개념에 따른 실시예들을 특정한 개시형태들에 대해 한정하려는 것이 아니며, 본 발명의 사상 및 기술 범위에 포함되는 변경, 균등물, 또는 대체물을 포함한다.

[0015] 제1 또는 제2 등의 용어를 다양한 구성요소들을 설명하는데 사용될 수 있지만, 상기 구성요소들은 상기 용어들에 의해 한정되어서는 안 된다. 상기 용어들은 하나의 구성요소를 다른 구성요소로부터 구별하는 목적으로만, 예를 들어 본 발명의 개념에 따른 권리 범위로부터 이탈되지 않은 채, 제1 구성요소는 제2 구성요소로 명명될 수 있고, 유사하게 제2 구성요소는 제1 구성요소로도 명명될 수 있다.

[0016] 어떤 구성요소가 다른 구성요소에 "연결되어" 있다거나 "접속되어" 있다고 언급된 때에는, 그 다른 구성요소에 직접적으로 연결되어 있거나 또는 접속되어 있을 수도 있지만, 중간에 다른 구성요소가 존재할 수도 있다고 이해되어야 할 것이다. 반면에, 어떤 구성요소가 다른 구성요소에 "직접 연결되어" 있다거나 "직접 접속되어" 있다고 언급된 때에는, 중간에 다른 구성요소가 존재하지 않는 것으로 이해되어야 할 것이다. 구성요소들 간의 관계를 설명하는 표현들, 예를 들어 "~사이에"와 "바로~사이에" 또는 "~에 직접 이웃하는" 등도 마찬가지로 해석되어야 한다.

[0017] 본 명세서에서 사용한 용어는 단지 특정한 실시예들을 설명하기 위해 사용된 것으로, 본 발명을 한정하려는 의도가 아니다. 단수의 표현은 문맥상 명백하게 다르게 뜻하지 않는 한, 복수의 표현을 포함한다. 본 명세서에서, "포함하다" 또는 "가지다" 등의 용어는 설시된 특징, 숫자, 단계, 동작, 구성요소, 부분품 또는 이들을 조합한 것이 존재함으로 지정하려는 것이지, 하나 또는 그 이상의 다른 특징들이나 숫자, 단계, 동작, 구성요소, 부분품 또는 이들을 조합한 것들의 존재 또는 부가 가능성을 미리 배제하지 않는 것으로 이해되어야 한다.

[0018] 다르게 정의되지 않는 한, 기술적이거나 과학적인 용어를 포함해서 여기서 사용되는 모든 용어들은 본 발명이 속하는 기술 분야에서 통상의 지식을 가진 자에 의해 일반적으로 이해되는 것과 동일한 의미를 가진다. 일반적으로 사용되는 사전에 정의되어 있는 것과 같은 용어들은 관련 기술의 문맥상 가지는 의미와 일치하는 의미를 갖는 것으로 해석되어야 하며, 본 명세서에서 명백하게 정의하지 않는 한, 이상적이거나 과도하게 형식적인 의미로 해석되지 않는다.

[0020] 이하, 실시예들을 첨부된 도면을 참조하여 상세하게 설명한다. 그러나, 특허출원의 범위가 이러한 실시예들에 의해 제한되거나 한정되는 것은 아니다. 각 도면에 제시된 동일한 참조 부호는 동일한 부재를 나타낸다.

[0022] 도 1은 일 실시예에 따른 컨테이너 클러스터 시스템의 구조를 나타낸 도면이다. 도 1을 참조하면, 컨테이너 클러스터 시스템(100)은 마스터(110) 및 노드(120)를 포함하는 적어도 하나의 노드를 포함한다. 컨테이너 클러스터 시스템(100)은 쿠버네티스(kubernetes)에 기반할 수 있다. 노드(120)는 팟(pod, 121)을 포함하며, 이와 같이 각 노드는 적어도 하나의 팟을 포함할 수 있다. 이하, 컨테이너 클러스터 시스템(100)을 마스터(110), 노드(120), 및 팟(121)을 기준으로 설명하지만, 아래의 설명은 다른 노드 및 다른 팟을 포함하는 컨테이너 클러스터 시스템(100)의 각 구성에도 적용될 수 있다.

[0023] 마스터(110)는 클러스터 전체를 관리하는 컨트롤러 역할을 한다. 마스터(110)는 어플리케이션을 스케줄링하거나, 어플리케이션의 항상성을 유지하거나, 어플리케이션을 스케일링하고, 새로운 변경 사항을 순서대로 반영하는 것과 같은 클러스터 내 활동을 조율할 수 있다.

[0024] 노드(120)는 컨테이너가 배포되는 머신(가상 머신 혹은 물리적인 서버)이다. 노드(120)는 어플리케이션을 구동하는 작업자라고 볼 수 있다. 노드(120)에는 쿠버렛(kuberlet)이라는 에이전트가 설치될 수 있고, 마스터(110)는 이를 통해 노드(120)를 관리할 수 있다. 노드(120)는 컨테이너의 운영을 담당하는 도커(docker)나 krt와 같은 툴을 가질 수도 있다.

[0025] 컨테이너 클러스터의 특징 중 하나는 컨테이너를 개별적으로 하나씩 배포하는 것이 아니라 팟(121) 단위로 배포하는 것인데, 팟(121)은 하나 이상의 컨테이너를 포함하는 컨테이터 클러스터의 기본 배포 단위이다. 팟(121)은 컨테이너 클러스터 시스템(100)의 서비스를 이용하는 각 사용자에게 할당될 수 있다. 예를 들어, 사용자와 팟(121)은 일대일로 매핑될 수 있다.

[0026] 또한, 컨테이너 클러스터의 구축은 오픈스택(openstack)을 기반으로 이뤄질 수 있다. 오픈스택은 KVM 등의 VM(Virtual Machine) 기반의 가상화 방식을 주로 사용하는데, 이를 컨테이너 기반으로 변경함으로써 인스턴스 생성에 소요되는 자원을 줄일 수 있다.

[0028] 도 2는 일 실시예에 따른 컨테이너 클러스터 시스템의 인증 구조를 나타낸 도면이다. 개발자나 운영자는 개발 및 관리를 위해 마스터 혹은 노드에 대해 접근 및 수정을 해야 하는데, 기존 구조에 따르면 이에 대한 권한 허가 및 인증이 마스터의 API 서버에서 이루어진다. 기존의 중앙 집중형 인증 방식에 따르면 마스터 내

API(application programming interface) 서버에 인증 트래픽이 집중될 수 있다. 실시예에 따르면 인증 처리를 위해 각 노드가 마스터 내 API 서버를 보완하는 분산형 인증 방식이 수행될 수 있고, 이에 따라 API 서버의 인증 트래픽을 크게 감소시킬 수 있다. 이하, 도 2를 참조하여 실시예에 따른 분산형 인증 방식을 설명한다.

[0029] 도 2를 참조하면, 사용자(210)는 마스터(221) 및 제1 노드(225)를 이용하여 컨테이너 클러스터 시스템(220)에 의해 제공되는 서비스를 이용하기 위한 인증을 수행할 수 있다. 사용자(210)는 개발자나 운영자를 포함할 수 있다. 본 명세서에서 사용자(210)는 사용자 단말이나 사용자 서버와 같이 실제 사용자에 의해 사용되는 임의의 디바이스를 나타낼 수 있다.

[0030] 마스터(221)는 로그인 처리부(222)를 포함할 수 있다. 예를 들어, 로그인 처리부(222)는 마스터(221)의 API 서버의 identity provider에 해당할 수 있다. 로그인 처리부(222)는 사용자(210)로부터 로그인 정보를 포함하는 로그인 요청을 수신하고, 로그인 요청에 응답하여 사용자(210)의 인증을 위한 인증 토큰을 사용자(210)에게 제공할 수 있다. 로그인 정보는 ID와 같은 식별 정보, 패스워드 등을 포함할 수 있다. 로그인 처리부(222)는 로그인 요청 시 사용자(210)가 제공한 로그인 정보 및 사용자(210)가 서비스 가입 시 제공한 계정 정보를 비교하여 사용자(210)의 로그인 요청을 처리할 수 있다.

[0031] 로그인에 성공 시, 로그인 처리부(222)는 사용자(210)에게 서비스 인증을 위한 토큰들을 제공할 수 있다. 예를 들어, 토큰들은 인증 토큰, ID 토큰, 리프레시 토큰 등을 포함할 수 있다. 인증 토큰은 사용자(210)를 위한 인증 정보에 대응할 수 있고, ID 토큰은 사용자(210)를 식별하기 위한 식별 정보에 대응할 수 있다. 인증 토큰은 일정한 만료 기한을 가질 수 있는데, 리프레시 토큰은 이러한 인증 토큰의 만료에 따라 발급될 수 있다.

[0032] 토큰이 발급되면, 사용자(210)는 제1 노드(225)의 인증 프록시(226) 및 인증 처리부(227)를 통해 서비스를 이용하기 위한 인증을 수행할 수 있다. 예를 들어, 인증 프록시(226)는 kube-proxy에 대응할 수 있고, 인증 처리부(227)는 catLedger에 대응할 수 있다. 본 명세서에서 로그인은 서비스로의 접속을 위한 것이고, 인증은 서비스를 이용하기 위한 것이다. 즉, 양자는 서로 구분되는 개념에 해당한다. 예를 들어, 마스터 혹은 노드에 대한 접근 및 수정 작업 등을 위해 로그인과는 별도의 사용자 인증이 요구된다.

[0033] 사용자(210)가 서비스에 가입 시 사용자(210)에게는 노드들 중 어느 하나에 존재하는 팟이 할당될 수 있고, 사용자(210)는 인증을 통해 해당 팟에 액세스하여 팟에 관한 수정 등의 처리를 수행할 수 있다. 컨테이너 클러스터 시스템(220)은 제1 노드(225) 이외에 다수의 노드들을 포함할 수 있으며, 아래에서는 설명의 편의를 위해 제1 노드(225)의 구조 및 동작이 설명되지만, 아래의 설명은 다른 노드들에도 적용될 수 있다. 즉, 각 노드는 인증 프록시(226) 및 인증 처리부(227)의 대응 구성들을 포함할 수 있고, 각 대응 구성들을 통해 자신과 연계된 사용자를 위한 인증을 처리할 수 있다.

[0034] 제1 노드(225)는 이러한 다수의 노드들 중 사용자(210)에게 할당된 팟을 보유한 노드에 대응될 수 있다. 서비스 가입 시 사용자(210)에게 제1 노드(225)의 인증 프록시(226)에 액세스하기 위한 인증 프록시(226)의 네트워크 주소가 제공될 수 있다. 예를 들어, 네트워크 주소는 IP 정보 또는 포트 정보를 포함할 수 있다. 사용자(210)는 인증 절차를 수행하기 위해 인증 프록시(226)의 네트워크 주소를 통해 인증 프록시(226)에 액세스할 수 있다.

[0035] 인증 프록시(226)는 사용자(210)의 액세스에 따라 사용자(210)로부터 인증 토큰을 수신하여 인증 처리부(227)에 전달할 수 있다. 인증 처리부(227)는 인증 프록시(226)로부터 인증 토큰을 전달받아 사용자(210)에 관한 인증을 처리하고, 인증 프록시(226)를 통해 사용자(210)에게 인증 처리 결과를 전송할 수 있다.

[0036] 일 실시예에 따르면, 인증을 위해 블록체인이 이용될 수 있다. 블록체인은 크게 퍼블릭 블록체인(public blockchain)과 프라이빗 블록체인(private blockchain)이 존재한다. 블록체인의 기본 원리는 관리 대상 데이터를 '블록'이라고 하는 소규모 데이터들이 P2P(peer to peer) 방식 기반으로 생성된 체인 형태의 분산 데이터 저장 환경에 저장되어, 누구라도 임의로 수정할 수 없고 누구나 변경의 결과를 열람할 수 있는 분산 컴퓨팅 기술 기반의 데이터 보호 기술이다.

[0037] 그러나 누구나 변경의 결과를 열람할 수 있다는 것에 대한 트레이드 오프로 트랜잭션 처리 속도에 큰 제약이 생긴다. 퍼블릭 블록체인은 PoW(Proof of Work), PoS(Proof of Stake) 등의 합의 알고리즘을 써야 누구라도 임의로 수정할 수 없다는 것을 보증할 수 있다. 따라서 합의에 시간이 걸리게 되고, 이에 따라 시간당 트랜잭션 처리 속도(transaction per second, TPS)가 급격히 낮아진다.

[0038] 프라이빗 블록체인은 합의에 아무나 참여할 수 없는 대신에 허용된 일부 기관 및 유저만이 합의에 참여할 수 있도록 설계되었다. 합의에 참여하는 피어가 한정되면서 합의 알고리즘들 중 빠른 처리 속도를 가진 알고리즘을

블록체인 시스템에 적용할 수 있게 되었다.

[0039] 대표적인 프라이빗 블록체인 시스템으로는 Hyperledger Fabric, Tendermint, Openchain 등이 있다. 이 중 Hyperledger Fabric은 PBFT(Practical Byzantine Fault Tolerance) 방식 등의 알고리즘을 도입하여 기존 비트코인에서의 7 TPS, 이더리움의 20TPS를, 1000 TPS 수준까지 높이기 위한 블록체인 시스템이다.

[0040] 인증 처리부(227)는 각 사용자의 인증 정보 및 인증 기록을 저장하는 블록체인 기반의 분산 원장을 이용하여 인증 토큰에 기초한 사용자(210)에 관한 인증을 처리할 수 있다. 분산 원장은 마스터(221) 및 제1 노드(225)를 비롯한 각 노드에 블록체인 기반으로 동기화되어 저장될 수 있다. 예를 들어, 분산 원장은 블록체인의 블록에 저장될 수 있다. 일 실시예에 따르면, 분산 원장에는 각 사용자의 인증 토큰 리스트가 저장될 수 있으며, 인증 처리부(227)는 인증 프록시(226)로부터 전달받은 인증 토큰과 분산 원장의 인증 토큰 리스트를 비교하여 사용자(210)에 관한 인증을 처리할 수 있다.

[0041] 일 실시예에 따르면, 인증 처리부(227)에 의해 이용되는 블록체인은 인증 프록시(226)를 포함하는 제한된 대상에게만 액세스를 허용하는 프라이빗 블록체인일 수 있다. 인증 프록시(226)는 인증된 제1 노드(225)의 구성이므로, 프라이빗 블록체인으로의 접근이 허용될 수 있다. 이와 같이 각 노드의 인증 프록시나 각 사용자는 프라이빗 블록체인으로의 액세스를 위한 권한을 보유할 수 있다.

[0042] 일 실시예에 따르면, 인증 처리부(227)에 의해 이용되는 프라이빗 블록체인은 PBFT 방식에 기반할 수 있다. 블록체인의 블록에 데이터 변경이 발생하면(예: 새로운 사용자의 가입에 따라 해당 사용자의 인증 토큰에 관한 정보가 블록에 추가되는 것) 데이터 변경에 관한 합의가 수행될 수 있는데, PBFT 방식의 경우 데이터 변경 시 비잔틴 장애를 허용하고 적어도 4개의 구성 노드들에 의한 투표에 기초하여 블록 내 데이터를 확정할 수 있다.

[0043] 예를 들어, 새로운 사용자의 가입에 따라 해당 사용자의 인증 정보가 추가되어 마스터(221)의 블록 데이터가 변경되면, 마스터(211)의 블록 데이터를 기준으로 각 노드의 블록 데이터가 동기화될 수 있다. 동기화가 완료되면 각 노드는 동기화가 완료되었음을 다른 노드들에게 알릴 수 있고, 전체 노드 중 일정 수나 일정 비율(예: 전체 노드의 2/3 이상)이 동기화된 데이터의 유효성에 관한 합의에 이르면, 해당 블록 데이터가 유효한 것으로 처리될 수 있다.

[0044] 상술된 것과 같이, 인증 프록시(226) 및 인증 처리부(227)에 의해 제1 노드(225)에서 사용자(210)의 인증 처리 중 적어도 일부가 수행될 수 있다. 즉, 이러한 인증 프록시(226) 및 인증 처리부(227)의 동작을 통해 마스터(221)의 인증 처리가 분산화될 수 있다.

[0045] 한편, 마스터(221), 로그인 처리부(222), 제1 노드(225), 인증 프록시(226), 및 인증 처리부(227)와 같이 본 명세서에 기술된 각 모듈(module)은 본 명세서에서 설명되는 각 명칭에 따른 기능과 동작을 수행할 수 있는 하드웨어를 의미할 수도 있고, 특정 기능과 동작을 수행할 수 있는 컴퓨터 프로그램 코드를 의미할 수도 있고, 또는 특정 기능과 동작을 수행시킬 수 있는 컴퓨터 프로그램 코드가 탑재된 전자적 기록 매체, 예를 들어 프로세서 또는 마이크로 프로세서를 의미할 수 있다. 다시 말해, 모듈이란 본 발명의 기술적 사상을 수행하기 위한 하드웨어 및/또는 상기 하드웨어를 구동하기 위한 소프트웨어의 기능적 및/또는 구조적 결합을 의미할 수 있다.

[0047] 도 3은 일 실시예에 따른 중앙 집중형 인증 방식의 컨테이너 클러스터 시스템을 나타낸 도면이고, 도 4는 일 실시예에 따른 분산형 인증 방식의 컨테이너 클러스터 시스템을 나타낸 도면이다. 도 3 및 도 4를 참조하면, 중앙 집중형 인증 방식의 컨테이너 클러스터 시스템에 비해 분산형 인증 방식의 컨테이너 클러스터 시스템은 catLedger를 더 포함한다. Kube-proxy는 상술된 인증 프록시에 해당할 수 있고, catLedger는 상술된 인증 처리부에 해당할 수 있다.

[0048] 중앙 집중형 인증 방식에 따르면 마스터 내 API 서버에서 사용자 인증이 처리되었으나, 분산형 인증 방식에 따르면 catLedger 및 Kube-proxy를 통해 사용자 인증이 분산되어 처리될 수 있다. 예를 들어, 각 사용자의 인증 정보(예: 인증 토큰)는 블록체인을 통해 마스터 및 노드들 각각에 동기화되어 저장될 수 있다. 마스터 및 노드들 각각의 catLedger는 각 사용자의 인증 정보를 블록 내 분산 원장에 저장하고 동기화할 수 있다. 사용자의 인증 요청이 있을 시, 사용자의 인증 토큰은 해당 사용자의 팟이 속한 노드의 Kube-proxy를 통해 해당 사용자의 팟이 속한 노드의 catLedger에 전달될 수 있고, 해당 catLedger는 블록체인 내 분산 원장을 이용하여 사용자 인증을 수행할 수 있다.

[0050] 도 5는 일 실시예에 따른 컨테이너 클러스터 시스템의 인증 방법을 나타낸 도면이다.

[0051] 도 5를 참조하면, 사용자(501)는 단계(510)에서 로그인 정보에 기초하여 로그인 처리부(502)에게 로그인을 요청

한다. 로그인 처리부(502)는 마스터 내 API 서버에 존재할 수 있으며, 예를 들어 API 서버 내 identity provider에 해당할 수 있다. 로그인에 성공 시, 로그인 처리부(502)는 단계(520)에서 사용자(501)에게 인증을 위한 토큰들을 전달한다.

[0052] 사용자(501)는 인증을 위해 단계(530)에서 인증 프록시(503)에 접근하여 인증 프록시(503)에게 인증 토큰을 전송하며, 인증 프록시(503)는 단계(540)에서 인증 처리부(504)에게 인증 토큰을 전달한다. 인증 처리부(504)는 단계(550)에서 사용자(501)를 위한 인증을 처리한다. 인증 처리부(504)는 블록체인 내 인증 토큰 리스트를 이용하여 사용자(501)를 위한 인증을 처리할 수 있다. 인증 프록시(503) 및 인증 처리부(504)는 노드 내에 존재할 수 있으며, 예를 들어 인증 프록시(503)는 노드 내 kube-proxy에 해당할 수 있고, 인증 처리부(504)는 노드 내 catLedger에 해당할 수 있다.

[0053] 인증 처리부(504)는 단계(560)에서 인증 프록시(503)에 인증 결과를 전송할 수 있고, 인증 프록시(503)는 단계(570)에서 사용자(501)에게 인증 결과를 전달할 수 있다. 이와 같은 프로세스를 통해 마스터의 API 서버 이외에 일반 노드들에도 인증에 대한 기능이 쉽게 추가될 수 있다.

[0054] 본 발명의 실시예에 따른 아키텍처를 통해 블록체인 기반의 인증 방식을 도입하게 되면, API 서버를 중앙 집중형 구조가 아닌 분산형 구조로 구현할 수 있다. 따라서 인증 요청이 서버의 수용량보다 많아지면 새로운 노드를 추가하거나 노드들의 성능을 높이는 것으로 해결된다. 이 방법들은 서비스를 일시정지하지 않아도 되는 방법이기 때문에 기존 방식보다 효과적이다.

[0055] 또한 인증 요청을 마스터의 API 서버에서만 처리하는 것이 아니고 일반 노드들에서도 처리할 수 있는 구조이기 때문에 접속하고자 하는 노드에서 인증을 처리할 수 있다. 따라서 인증 트래픽이 중앙 서버를 거쳤다가 다시 해당 노드로 가지 않아도 된다. 즉, 인증 과정에서 발생하는 추가 트래픽을 줄일 수 있기 때문에, 제안된 아키텍처를 사용하면 네트워크 트래픽 병목 현상 등 인증과정에서 네트워크 지연을 유발하는 요소를 적절히 방지할 수 있다.

[0057] 이상에서 설명된 장치는 하드웨어 구성요소, 소프트웨어 구성요소, 및/또는 하드웨어 구성요소 및 소프트웨어 구성요소의 조합으로 구현될 수 있다. 예를 들어, 실시예들에서 설명된 장치 및 구성요소는, 예를 들어, 프로세서, 콘트롤러, ALU(Arithmetic Logic Unit), 디지털 신호 프로세서(digital signal processor), 마이크로컴퓨터, FPGA(Field Programmable Gate Array), PLU(Programmable Logic Unit), 마이크로프로세서, 또는 명령(instruction)을 실행하고 응답할 수 있는 다른 어떠한 장치와 같이, 하나 이상의 범용 컴퓨터 또는 특수 목적 컴퓨터를 이용하여 구현될 수 있다. 처리 장치는 운영 체제(OS) 및 상기 운영 체제 상에서 수행되는 하나 이상의 소프트웨어 어플리케이션을 수행할 수 있다. 또한, 처리 장치는 소프트웨어의 실행에 응답하여, 데이터를 접근, 저장, 조작, 처리 및 생성할 수도 있다. 이해의 편의를 위하여, 처리 장치는 하나가 사용되는 것으로 설명된 경우도 있지만, 해당 기술분야에서 통상의 지식을 가진 자는, 처리 장치가 복수 개의 처리 요소(processing element) 및/또는 복수 유형의 처리 요소를 포함할 수 있음을 알 수 있다. 예를 들어, 처리 장치는 복수 개의 프로세서 또는 하나의 프로세서 및 하나의 콘트롤러를 포함할 수 있다. 또한, 병렬 프로세서(parallel processor)와 같은, 다른 처리 구성(processing configuration)도 가능하다.

[0058] 소프트웨어는 컴퓨터 프로그램(computer program), 코드(code), 명령(instruction), 또는 이들 중 하나 이상의 조합을 포함할 수 있으며, 원하는 대로 동작하도록 처리 장치를 구성하거나 독립적으로 또는 결합적으로(collectively) 처리 장치를 명령할 수 있다. 소프트웨어 및/또는 데이터는, 처리 장치에 의하여 해석되거나 처리 장치에 명령 또는 데이터를 제공하기 위하여, 어떤 유형의 기계, 구성요소(component), 물리적 장치, 가상 장치(virtual equipment), 컴퓨터 저장 매체 또는 장치, 또는 전송되는 신호 파(signal wave)에 영구적으로, 또는 일시적으로 구체화(embody)될 수 있다. 소프트웨어는 네트워크로 연결된 컴퓨터 시스템 상에 분산되어서, 분산된 방법으로 저장되거나 실행될 수도 있다. 소프트웨어 및 데이터는 하나 이상의 컴퓨터 판독 가능 기록 매체에 저장될 수 있다.

[0059] 실시예에 따른 방법은 다양한 컴퓨터 수단을 통하여 수행될 수 있는 프로그램 명령 형태로 구현되어 컴퓨터 판독 가능 매체에 기록될 수 있다. 상기 컴퓨터 판독 가능 매체는 프로그램 명령, 데이터 파일, 데이터 구조 등을 단독으로 또는 조합하여 포함할 수 있다. 상기 매체에 기록되는 프로그램 명령은 실시예를 위하여 특별히 설계되고 구성된 것들이거나 컴퓨터 소프트웨어 당업자에게 공지되어 사용 가능한 것일 수도 있다. 컴퓨터 판독 가능 기록 매체의 예에는 하드 디스크, 플로피 디스크 및 자기 테이프와 같은 자기 매체(magnetic media), CD-ROM, DVD와 같은 광기록 매체(optical media), 플롭티컬 디스크(floptical disk)와 같은 자기-광 매체(magneto-optical media), 및 롬(ROM), 램(RAM), 플래시 메모리 등과 같은 프로그램 명령을 저장하고 수행하도

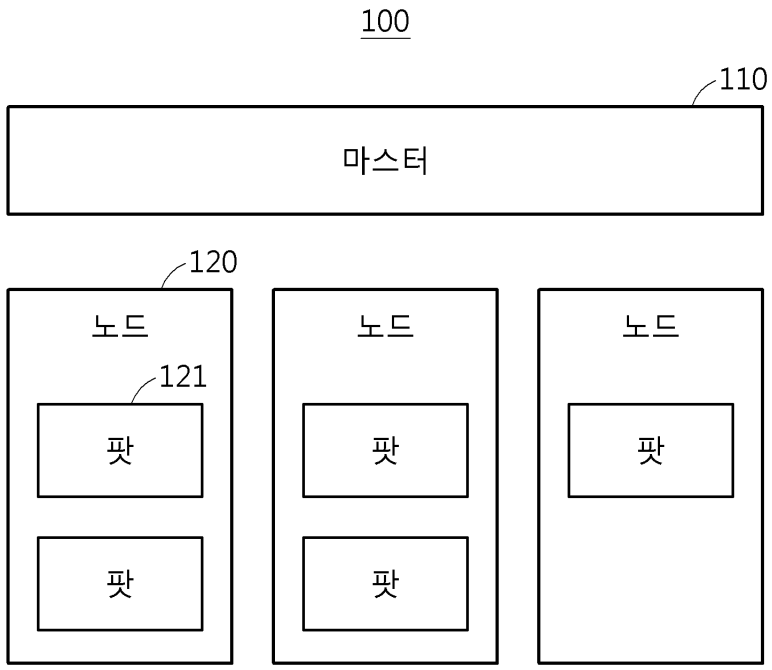
록 특별히 구성된 하드웨어 장치가 포함된다. 프로그램 명령의 예에는 컴파일러에 의해 만들어지는 것과 같은 기계어 코드뿐만 아니라 인터프리터 등을 사용해서 컴퓨터에 의해서 실행될 수 있는 고급 언어 코드를 포함한다. 상기된 하드웨어 장치는 실시예의 동작을 수행하기 위해 하나 이상의 소프트웨어 모듈로서 작동하도록 구성될 수 있으며, 그 역도 마찬가지이다.

[0060] 이상과 같이 실시예들이 비록 한정된 실시예와 도면에 의해 설명되었으나, 해당 기술분야에서 통상의 지식을 가진 자라면 상기의 기재로부터 다양한 수정 및 변형이 가능하다. 예를 들어, 설명된 기술들이 설명된 방법과 다른 순서로 수행되거나, 및/또는 설명된 시스템, 구조, 장치, 회로 등의 구성요소들이 설명된 방법과 다른 형태로 결합 또는 조합되거나, 다른 구성요소 또는 균등물에 의하여 대치되거나 치환되더라도 적절한 결과가 달성될 수 있다.

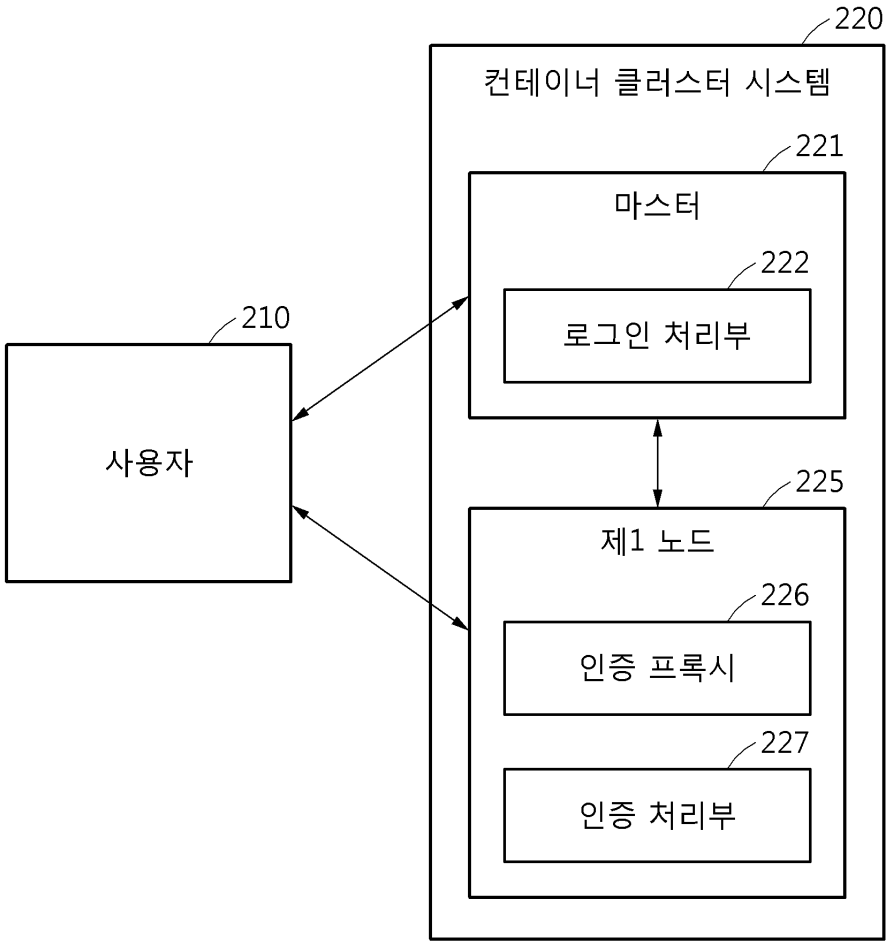
[0061] 그러므로, 다른 구현들, 다른 실시예들 및 특허청구범위와 균등한 것들도 후술하는 특허청구범위의 범위에 속한다.

도면

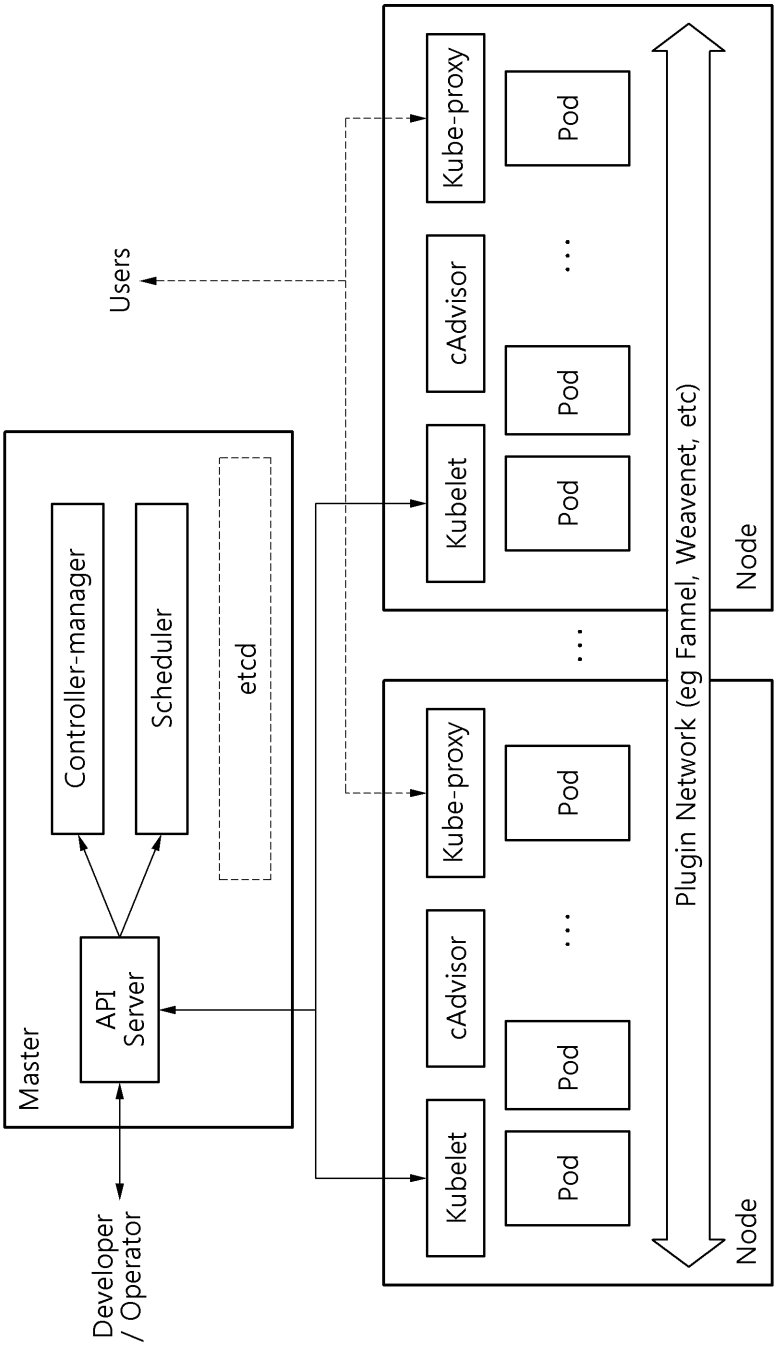
도면1



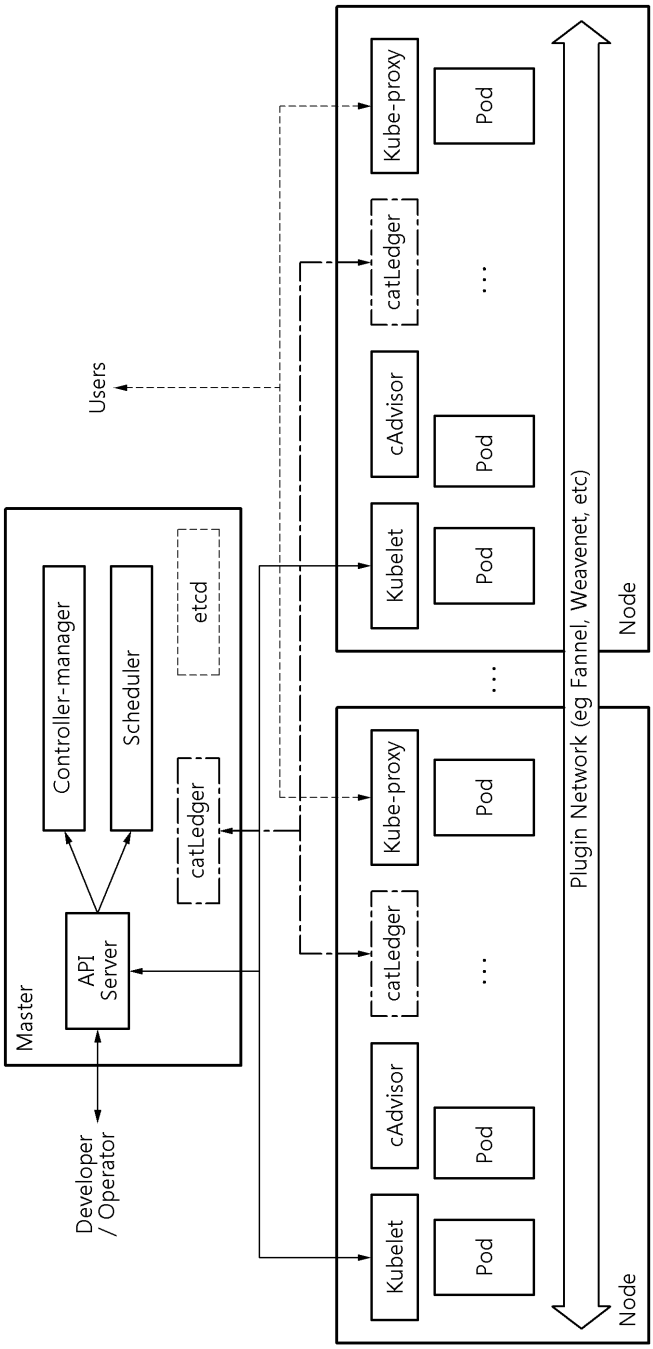
도면2


220
컨테이너 클러스터 시스템
221
마스터
222
로그인 처리부
210
사용자
225
제1 노드
226
인증 프록시
227
인증 처리부

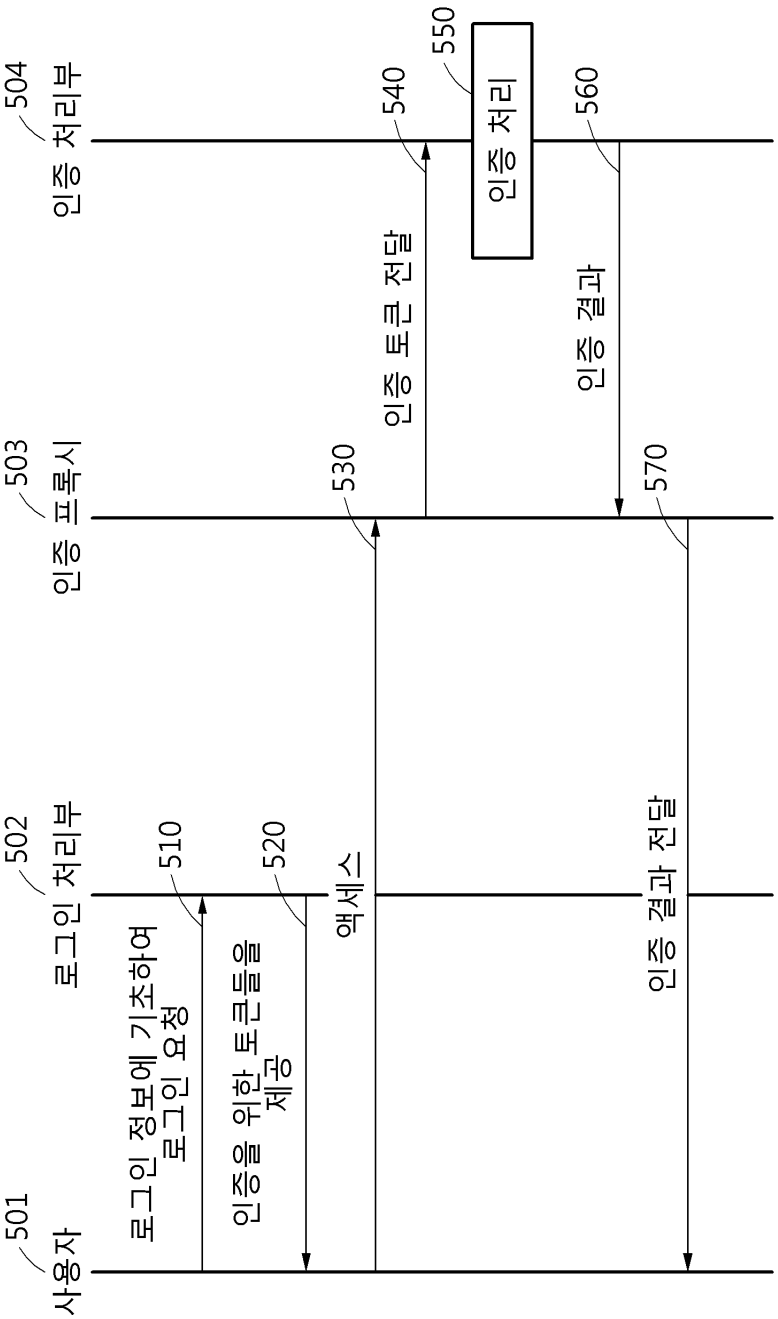
도면3


Users
Master
Controller-manager
Scheduler
etcd
API Server
Developer / Operator
Kubelet
cAdvisor
Kube-proxy
Pod
Node
Plugin Network (eg Fannel, Weavenet, etc)

도면4


Users
Master
Controller-manager
Scheduler
etcd
API Server
catLedger
Developer / Operator
Kubelet
cAdvisor
catLedger
Kube-proxy
Pod
Node
Plugin Network (eg Fannel, Weavenet, etc)

도면5


501
사용자
502
로그인 처리부
503
인증 프록시
504
인증 처리부
로그인 정보에 기초하여 로그인을 요청
510
인증을 위한 토큰들을 제공
520
액세스
530
인증 토큰 전달
540
인증 처리
550
인증 결과
560
인증 결과 전달
570