Blockchain Infrastructure

시스템/네트워크 운영자 대상

(for IT Pros and System Administrators)

2019. 6.

안종석
james@jslab.kr

- 사용 유효기간을 2019년 12월 까지로 권합니다.
- 실습 교재는 별도 입니다.

JS Lab

1

목차

I. 개요
II. 퍼블릭 블록체인 (Public Blockchain)
III. 스마트 계약 (Smart Contract)
IV. 프라이빗 블록체인 (Private Blockchain)
V. 컨소시엄 블록체인 (Consortium Blockchain)
VI. 하이퍼레저 (Hyperledger)
VII. 컨센서스 (Consensus)
VIII. 사례 (Use Cases)
❖ **실습** (별도 교재)

2

I. **개요**

II. **퍼블릭 블록체인** (Public Blockchain)

III. **스마트 계약** (Smart Contract)

IV. **프라이빗 블록체인** (Private Blockchain)

V. **컨소시엄 블록체인** (Consortium Blockchain)

VI. **하이퍼레저** (Hyperledger)

VII. **컨센서스** (Consensus)

VIII. **사례** (Use Cases)

❖ **실습** (별도 교재)

james@jslab.kr

I. 개요

❖ **사토시 나카모토가 메일로 'Bitcoin' white paper 발표** (2008년 10월 31일)

- **블록체인 백서** (http://www.bitcoin.org/bitcoin.pdf)
- **"저는 신뢰할 만한 전자화폐(electronic cash) 시스템을 연구해오고 있습니다. 그 시스템은 완전한 개인 간 거래가 가능하며 신뢰할 만한 제3의 기관이 필요 없습니다."**

From: Satoshi Nakamoto <satoshi <at> vistomail.com>
Subject: **Bitcoin P2P e-cash paper**
Newsgroups: **gmane.comp.encryption.general**
Date: 2008-10-31 18:10:00 GMT (4 years, 52 weeks, 1 day, 3 hours and 23 minutes ago)

I've been working on a new electronic cash system that's fully peer-to-peer, with no trusted third party.

The paper is available at:
http://www.bitcoin.org/bitcoin.pdf

The main properties:
Double-spending is prevented with a peer-to-peer network.
No mint or other trusted parties.
Participants can be anonymous.
New coins are made from Hashcash style proof-of-work.
The proof-of-work for new coin generation also powers the network to prevent double-spending.

Bitcoin: A Peer-to-Peer Electronic Cash System

Abstract. A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without the burdens of going through a financial institution. Digital signatures provide part of the solution, but the main benefits are lost if a trusted party is still required to prevent double-spending. We propose a solution to the double-spending problem using a peer-to-peer network. The network timestamps transactions by hashing them into an ongoing chain of hash-based proof-of-work, forming a record that cannot be changed without redoing the proof-of-work. The longest chain not only serves as proof of the sequence of events witnessed, but proof that it came from the largest pool of CPU power. As long as honest nodes control the most CPU power on the network, they can generate the longest chain and outpace any attackers. The network itself requires minimal structure. Messages are broadcasted on a best effort basis, and nodes can leave and rejoin the network at will, accepting the longest proof-of-work chain as proof of what happened while they were gone.

2008년 9월 15일 파산한 미국 최대의 투자은행 리먼 브라더스와 그로 인한 금융 위기에 대한 비판 배경

http://www.bitcoin.org/bitcoin.pdf

james@jslab.kr

JS Lab

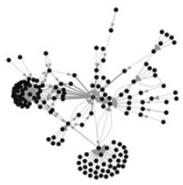
I. 개요

❖ **비트코인 탄생** (2009년 1월 3일)

- **사토시 나카모토는 최초 노드인 자신의 컴퓨터에서 최초의 블록 '제네시스 블록(genesis block)' 생성**
- **암호 전문가인 할 피니(Hal Finny)는 사토시 나카모토와 협력하며 비트코인 시스템을 개선하고 2번째 노드를 생성**
- **사토시 나카모토가 할 피니에게 10 비트코인 전송** (1월 9일 최초의 비트코인 거래)
- **비트코인으로 프로그래머 한예츠(Laszlo Hanyecz)가 1만 비트코인으로 30달러 가격의 피자 2판을 주문** (2010년 5월)

https://coinmarketcap.com/

Transaction visualization



블록체인은 개인키/공개키 방식으로 정보가 누구의 것인지는 알 수 없지만 거래 내역을 역추적해보면 누구인지 알 수 있는 경우가 있음.

https://cloud.google.com/blog/products/gcp/bitcoin-in-bigquery-blockchain-analytics-on-public-data

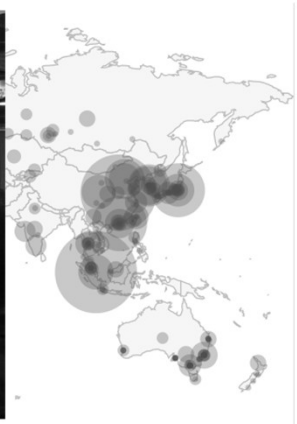
james@jslab.kr

5

I. 개요

❖ **2019년 현재 10,000여개의 비트코인 Network Node**

- **9677 reachable nodes in the Bitcoin network as of Fri May 31 2019**
  (https://bitnodes.earn.com/nodes/)



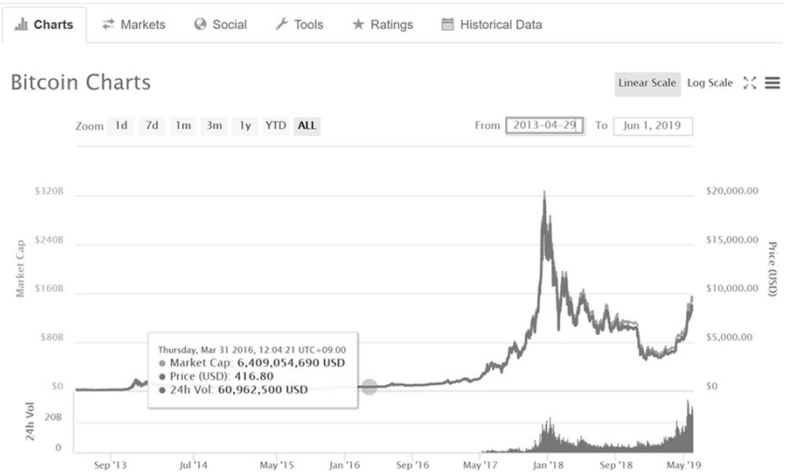
https://bitnodes.earn.com/nodes/

james@jslab.kr

6

I. 개요

❖ **Market**



https://coinmarketcap.com/currencies/bitcoin/#charts

james@jslab.kr

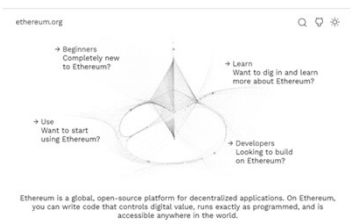
7

I. 개요

❖ **비트코인은 화폐 가치 저장/전달 기능만 제공** (1세대 블록체인)

❖ **블록체인 기술을 가상통화 거래 이외의 목적으로 활용 시작** (2세대 블록체인)

- **러시아 출신 캐나다의 프로그래머 비탈릭 부테린(Vitalik Buterin)은 '차세대 스마트 계약과 탈중앙화된 어플리케이션 플랫폼(A Next Generation Smart Contract and Decentralized Application Platform)'이라는 제목의 이더리움(Ethereum) 백서를 발표하고, 다음 해인 2014년 이더리움 재단 설립**
- **조건을 충족하면 다음 절차를 코드로 자동으로 분산처리 수행** (스마트 계약)
- **분산 어플리케이션**(Decentralized Application, DApp)**을 개발/실행 가능**



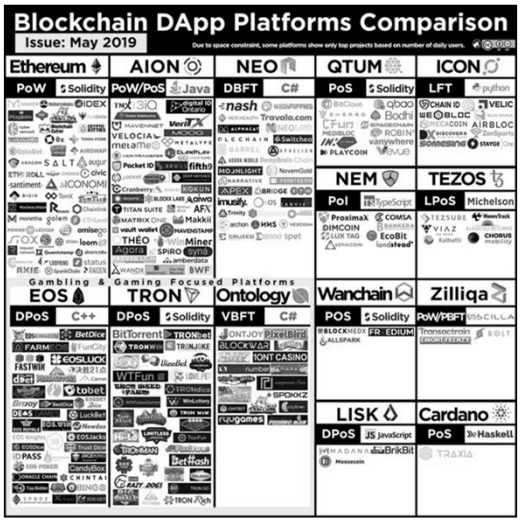
https://bitcoin.org/ko/ https://ethereum.org/

james@jslab.kr

8

I. 개요

❖ **Smart Contract DApp Platforms Comparison** (MAY 2019 UPDATE)

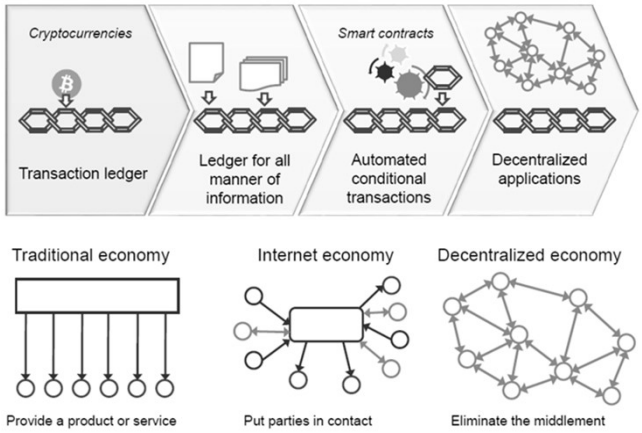


https://www.reddit.com/r/cardano/comments/bpvcoa/smart_contract_dapp_platforms_comparison_may_2019/

I. 개요

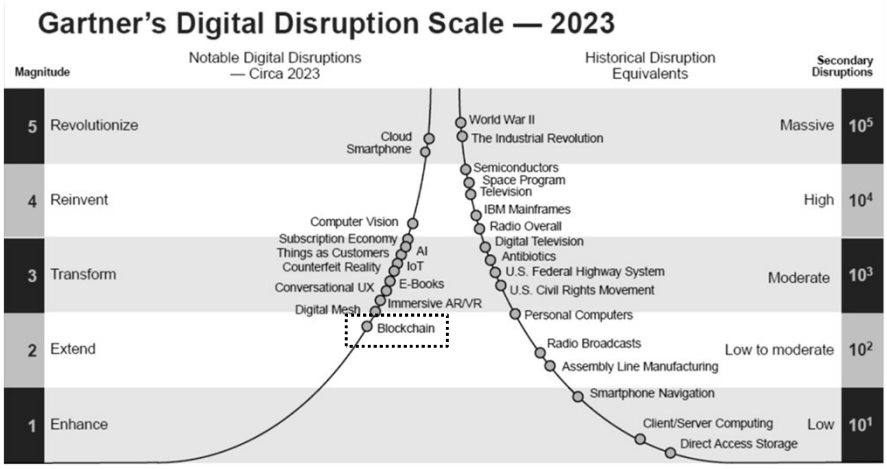
❖ **IDATE DigiWorld, Blockchain** (October 2016)

- **Blockchain Long Term Vision - disruptive vision**



I. 개요

❖ 가트너의 Digital Disruption Scale 전망



11

I. 개요

❖ 시장: 금융 분야(예)

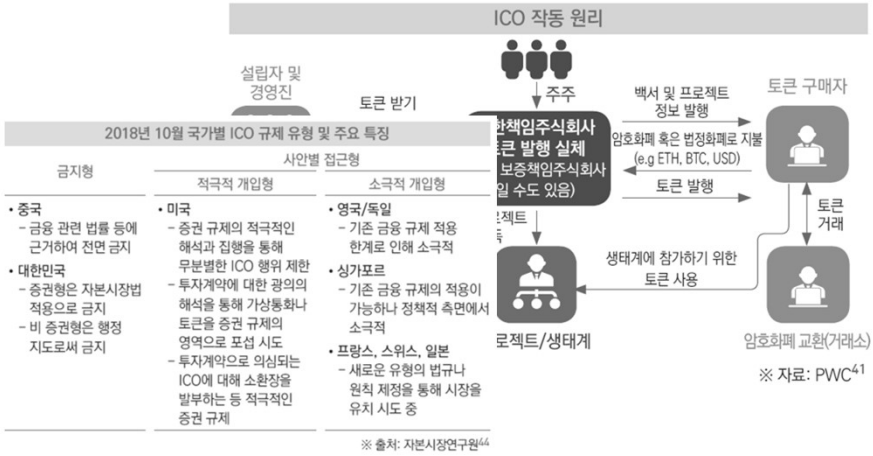
- 많은 기관들이 직접 블록체인 연구개발에 참여하여 가상통화를 개발하거나 코다, 하이퍼레저 패브릭(Hyperledger Fabric) 등 컨소시엄 블록체인 도입
- 골드만삭스는 세틀코인(SETLcoin), 시티그룹은 시티코인(Citicoin)을 개발
- 스코틀랜드 은행(Royal Bank of Scotland)은 코다를 도입하여 주택담보대출 납부 처리를 자동화블록체인 기술을 가상통화 거래 이외의 목적으로 활용 시작
- 미국 나스닥에서는 2015년 10월부터 블록체인 기반의 시스템 '링크'를 통해 비상장 주식을 거래 (기존 2~3일 소요 시간을 10분 이내로 단축)
- 카카오페이는 2017년 6월 공인인증서를 대체할 수 있는 블록체인 기반 인증서비스를 상용화
- 규제 샌드박스

12

I. 개요

❖ **ICO** (Initial Coin Offering)

- **코인:** 자체 블록체인 플랫폼에서 발행된 가상통화는 코인
- **토큰:** 블록체인 상 디앱에서 발행된 가상통화는 토큰

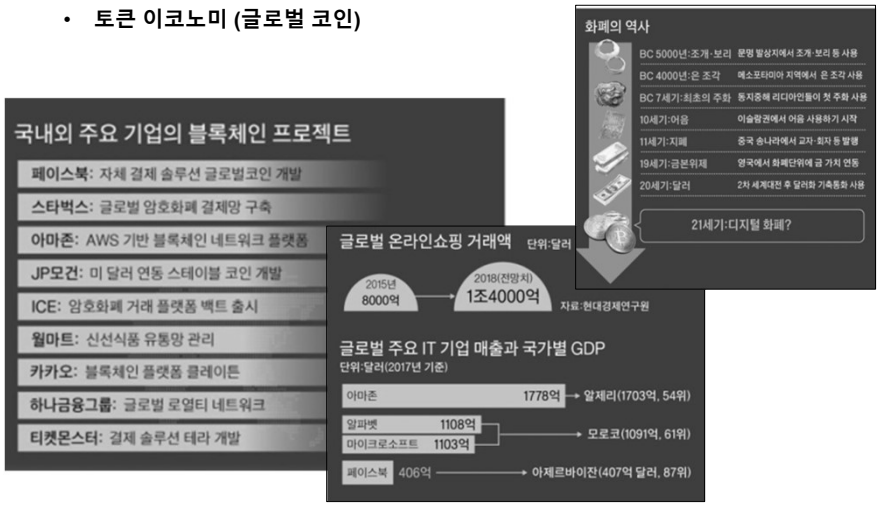


13

I. 개요

❖ **글로벌 시장의 변화**

- **토큰 이코노미 (글로벌 코인)**



https://m.news.naver.com/read.nhn?mode=LSD&sid1=001&oid=353&aid=0000034225

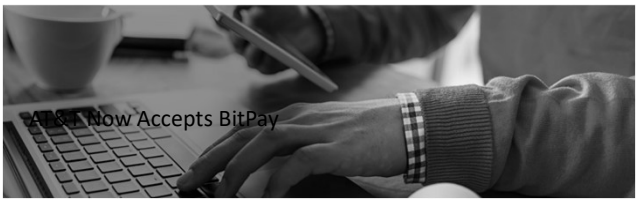
14

I. 개요

❖ **Telecom Giant AT&T Gains First-Mover Status by Accepting Crypto**

- **Customers Can Now Use BitPay to Pay Online Bills**

AT&T* will now accept online bill payments through BitPay, a respected cryptocurrency payment processor. AT&T becomes the first major U.S. mobile carrier to provide a cryptocurrency payment option to customers.
"We're always looking for ways to improve and expand our services," said Kevin McDorman, vice president, AT&T Communications Finance Business Operations. "We have customers who use cryptocurrency, and we are happy we can offer them a way to pay their bills with the method they prefer."
Customers will be able to select BitPay as a payment option when they log on to their accounts online or with the myAT&T app.



https://about.att.com/story/2019/att_bitpay.html

james@jslab.kr

15

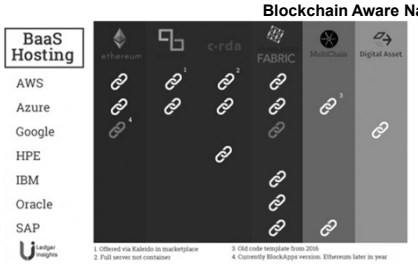
I. 개요

❖ **암호화폐 거래를 위한 블록체인 플랫폼**

❖ **Managed Blockchain Platform** (BaaS)

지불 게이트웨이 (Payment gateway) | 지갑 (Wallet) | 거래 (Exchange) | 블록체인 (Blockchain) | 중계 소프트웨어 (Arbitrage Software)

Blockchain Aware Native Scheduling (BANS)



https://etherscan.io/

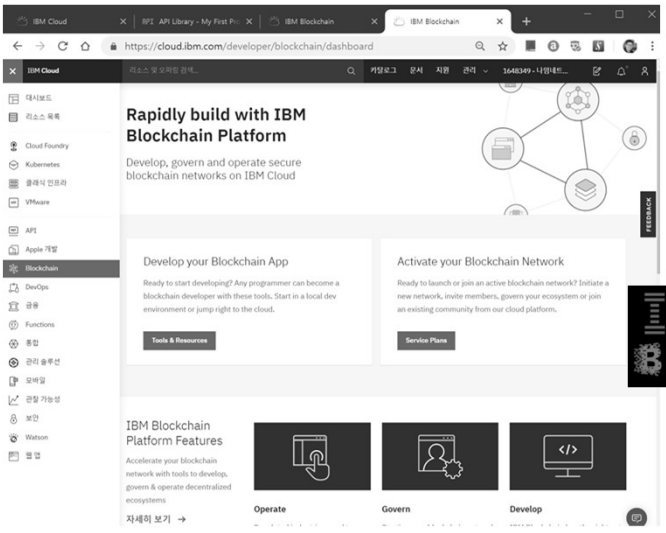
james@jslab.kr

16

james@jslab.kr

I. 개요

❖ **Blockchain @ Public Cloud: IBM Blockchain Platform**

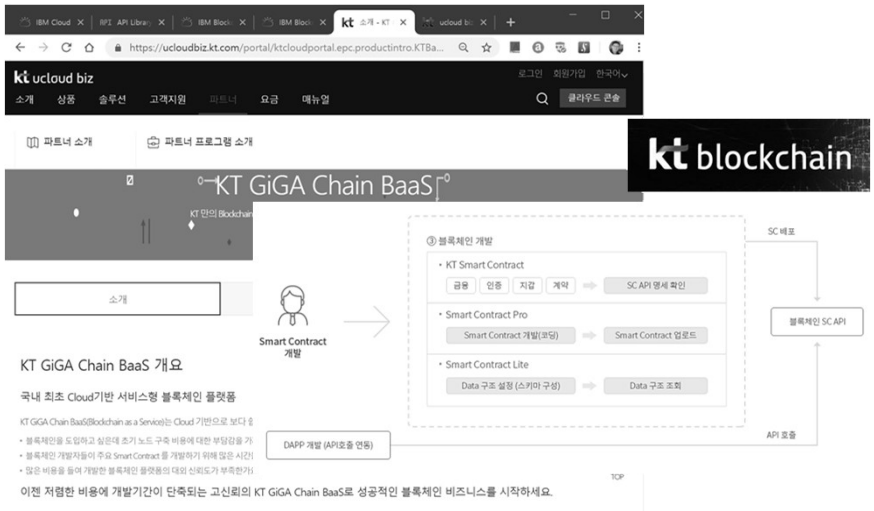


17

james@jslab.kr

I. 개요

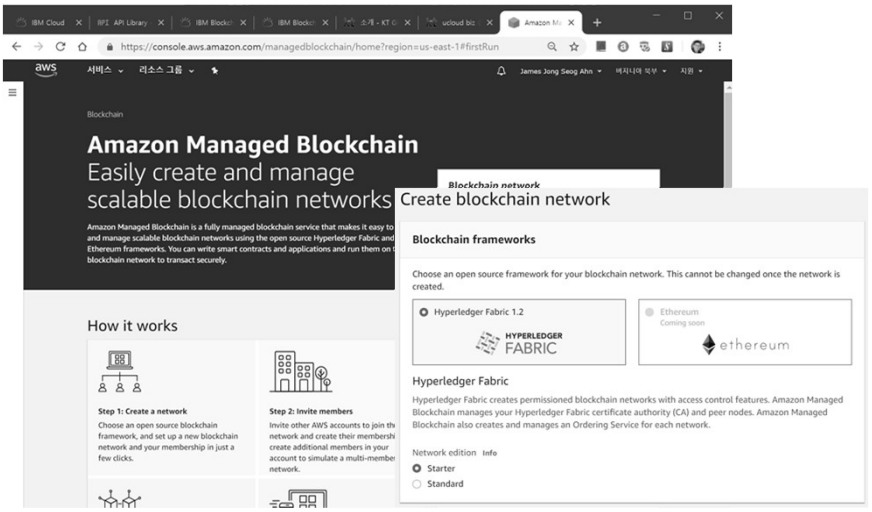
❖ **Blockchain @ Public Cloud: kt ucloud**



18

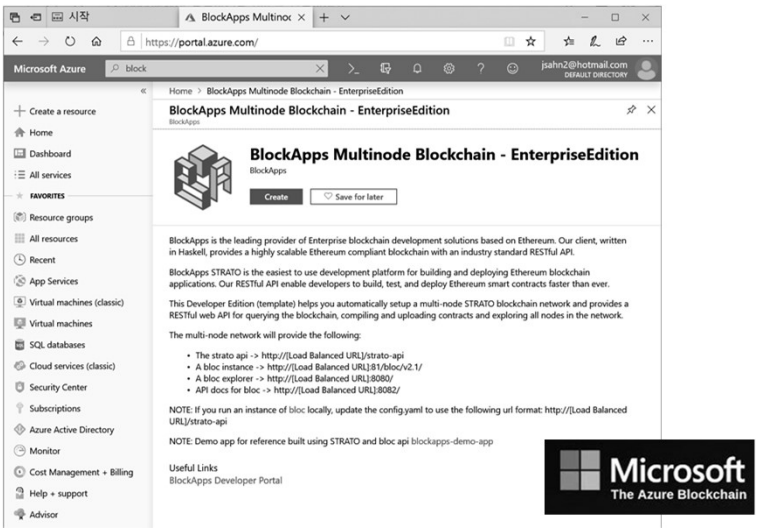
I. 개요

❖ **Blockchain @ Public Cloud: AWS**



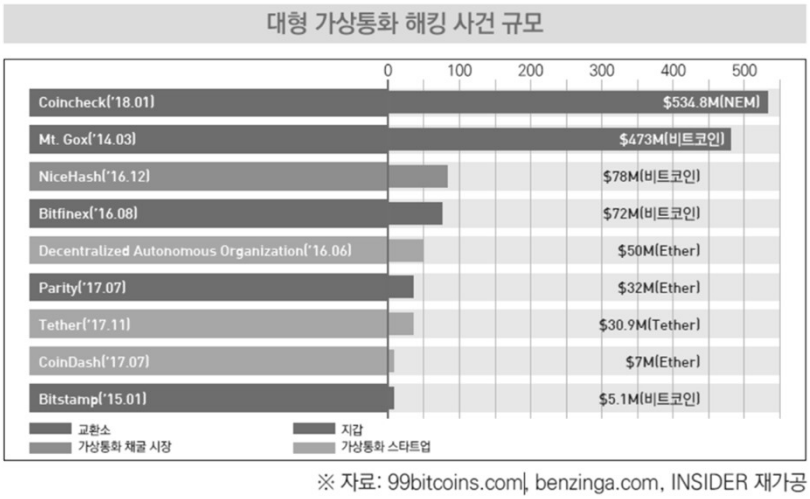
I. 개요

❖ **Blockchain @ Public Cloud: Microsoft Azure**



I. 개요

❖ 가상통화 해킹

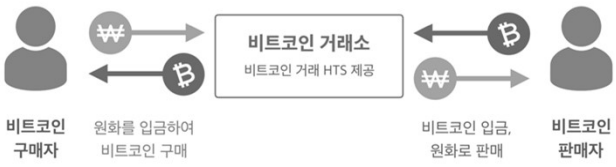


2018년 기술영향평가 결과보고 '블록체인의 미래' 과학기술정보통신부 · 한국과학기술기획평가원(KISTEP)

I. 개요

❖ 블록체인과 암호화폐 거래소

- **Pre-Trade → Trade → Post-Trade → Custody & securities servicing**
  - ✓ 비트코인 거래소는 비트코인 구매/판매를 위한 플랫폼 역할
  - ✓ 웹 HTS 호가창을 통해 시장가/지정가에 비트코인을 사고 팔 수 있음
  - ✓ 원화를 입금해서 비트코인을 구매 가능
  - ✓ 비트코인을 판매해서 원화를 출금 가능



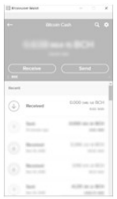
coinone

http://www.agiboo.com/blockchain-commodity-trading/blockchain-infographics-mg/

I. 개요

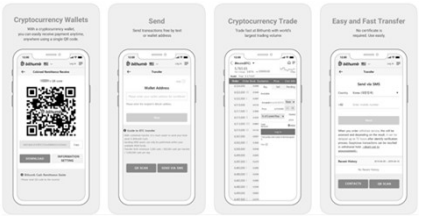
❖ 블록체인 기술 특징

- 거래이력 정보를 네트워크에 참가하는 전원에게 분산하여 보관 유지하고 참가자들의 합의를 통해 거래 데이터의 정당성을 보증하는 시스템
- 모두가 결과를 열람 가능
- 중계의 복잡성 제거와 탈이해 관계의 데이터의 집중화 가능
- 분산 원장 기술 사용
- P2P 방식 기반의 체인형태로 연결



비트코인 지갑(Wallet)

- 블록체인 기술 사용
  비트코인 – 평균 10분 당 하나씩 블록 생성



암호화폐 거래소 지갑(Wallet)

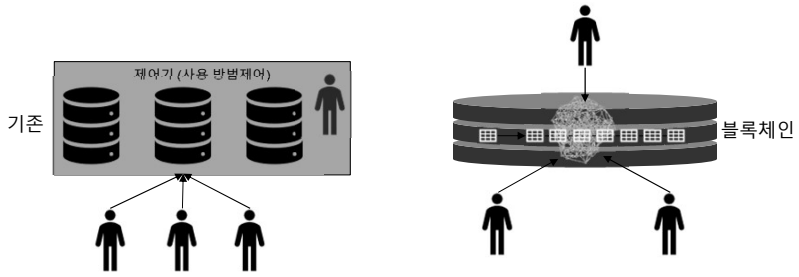
https://www.blockchain.com/explorer

james@jslab.kr

I. 개요

❖ 블록체인 기술의 기존 데이터베이스와 차이점

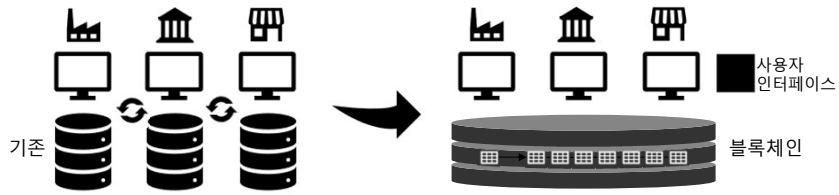
- 블록체인은 쓰기만 가능한 데이터 구조
- 새로운 기재(Entry)는 원장의 끝에 붙임
- 블록체인 내에서는 데이터의 편집이나 제거를 허락하는 관리자 권한이 없음
- 기존 장부에는 수표나 영수증 또는 약속어음의 교환내역이 기록되는 반면에, 블록체인은 그것 자체가 거래장부인 동시에 거래증서(수표, 영수증, 약속어음)



james@jslab.kr

I. 개요

❖ **블록체인(Blockchain):** 데이터가 안전하게 저장되고 감사 가능한 공유 데이터 계층을 생성하여 사용자 인터페이스를 제공



사일로에 저장한 데이터로 호환에 제한적 (검증 받은 인터페이스 필요)

안전하게 저장되어 감사(Audit) 가능한 공유(Shared) 데이터

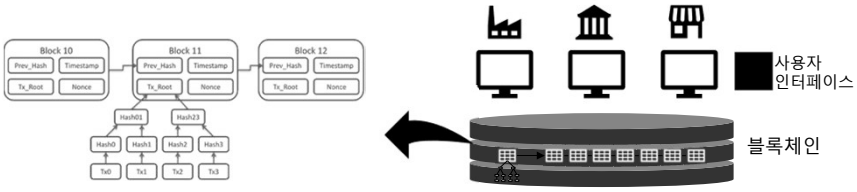
블록을 체인하여 구성

james@jslab.kr

25

I. 개요

❖ **블록(Block):** 특정 시간 동안 발생한 거래(Transaction)들을 처리한 리스트
❖ **체인(Chain):** 각 블록에 생성시각, 거래내역 묶음 해시, 해시 값 생성한 계산 횟수, 전 블록의 블록 해시 등을 지정하고 암호화 알고리즘을 사용하여 시간 순으로 블록을 연결
❖ **채굴(Mining):** 작업증명 PoW(Proof of Working)으로 블록을 생성하여 보상으로 가상통화를 받는 행위



Merkle Chains 또는 Merkle Trees
(효율적 저장)

해시 함수(hash function)는 임의의 길이의 데이터를 고정된 길이의 데이터로 매핑하는 함수이다. 해시 함수에 의해 얻어지는 값은 해시 값, 해시 코드, 해시 체크섬 또는 간단하게 해시라고 한다.

- **비트코인(Bitcoin)은 약 10분 단위로 전세계 2000여 개의 거래들을 모아 조건을 만족하는 해시(Hash) 값을 구해 새로운 블록을 생성(채굴)**
- **이더리움(Ethereum)은 12~15초마다 새로운 블록 생성**

https://en.wikipedia.org/wiki/Merkle_tree#/media/File:Hash_Tree.svg

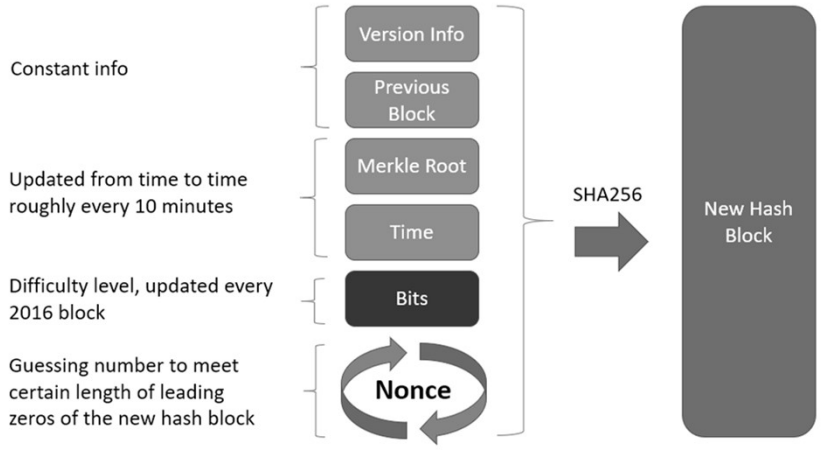
https://www.blockchain.com/ko/explorer

james@jslab.kr

26

I. 개요

❖ **Bitcoin Block Hashing**



https://themoneymongers.com/bitcoin-nonce/

27

I. 개요

❖ **Block Difficulty for Bitcoin**

- **Target(T) is a 32 byte(256 bit) number**
  - ✓ Maximum target(Tmax): 0x1d00ffff = 0x00000000FFFF0000000000000000000000000000000000000000000000000000
- **Difficulty(D): A measure of how difficult it is to find a new block compared to the easiest.**

  D = $Tmax$ **/** $Tcur$ **if** $\boldsymbol{Tcur}$ **= 0 , D = infinity**
- **Maximum Difficulty is set when** $\boldsymbol{Tcur}$ **= 1 ,** $Tmax$ **is defined as 0x1d00ffff**
- **예: If** $\boldsymbol{Tcur}$ **= 0x1b0404cb, then**

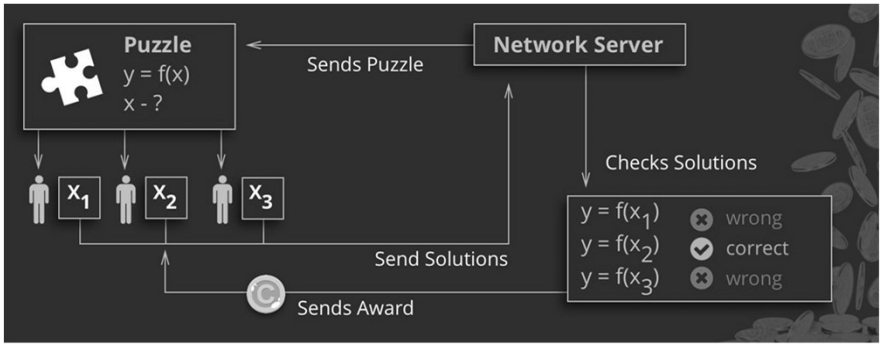
  D = $Tmax$ **0x1d00ffff /** $Tcur$ **0x1b0404cb = ? =16307.669773817162**
  - ✓ **Desired rate of finding a block: 1 block / 10 minutes.**
  - ✓ **Difficulty is adjusted after every 6*24*14=2016 blocks to maintain the rate.**

https://www.blockchain.com/en/charts/hash-rate?timespan=all

28

I. 개요

❖ **채굴: Mining = finding a nonce value (integer)**



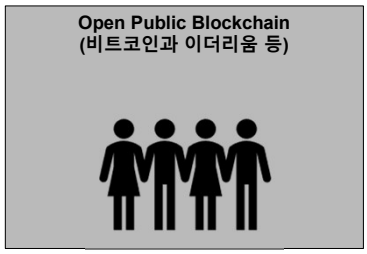
논스(Number used Once):블록체인에서 사용하기 위해 한번만 사용되는 임의의 수로 1씩 증가하는 카운트와 같은 역할을 하는 수이자, 일정 조건의 해시 값을 출력하기 위해 데이터와 함께 해시함수에 넣는 수다. 해시함수는 출력값(해시값)을 통해서 입력값(원래 데이터)을 복원할 수 없는 일방향성을 가지므로, 출력값이 갖춰야 할 조건을 만족시키기 위해서는 논스를 통해 해쉬의 입력값을 계속 바꿔야 한다. 논스가 1씩 증가해도 출력값을 놓고 볼 때는 임의의 수를 넣는 것과 같은 결과가 나옴

https://cointelegraph.com/explained/proof-of-work-explained

I. 개요

❖ **블록체인의 형태**

- **블록체인은 참여 허락이 필요 없는 permissionless**(Bitcoin 과 Ethereum)**와 허락이 필요한 permissioned** (Hyperledger blockchain frameworks) **블록체인이 있음**



JS Lab

I. 개요

❖ **블록체인의 종류:** 참여자 범위에 따라 퍼블릭, 프라이빗, 컨소시엄 3가지로 구분

- **퍼블릭 블록체인(Public Blockchain)**
- **프라이빗 블록체인(Private Blockchain)**
- **컨소시엄 블록체인(Consortium Blockchain)**

요소	퍼블릭 (Public)	프라이빗 (Private)	컨소시엄 (Consortium)
관리주체	모든 거래 참여자(탈중앙화)	한 중앙기관이 모든 권한 보유	컨소시엄에 소속된 참여자
거버넌스	한번 정해진 법칙을 바꾸기 매우 어려움	중앙기관의 의사결정에 따라 용이하게 법칙을 바꿀 수 있음	컨소시엄 참여자들의 합의에 따라 상대적으로 용이하게 법칙을 바꿀 수 있음
거래속도	네트워크 확장이 어렵고 거래 속도가 느림	네트워크 확장이 매우 쉽고 거래 속도가 빠름	네트워크 확장이 쉽고 거래 속도가 빠름
데이터 접근	누구나 접근 가능	허가받은 사용자만 접근가능	허가받은 사용자만 접근가능
식별성	익명성	식별 가능	식별 가능
거래증명	검증 알고리즘에 따라 거래 증명자가 결정되며 거래 증명자가 누구인지 사전에 알 수 없음	중앙기관에 의하여 거래증명이 이루어짐	거래 증명자가 인증을 거쳐 알려진 상태이며 사전 합의된 규칙에 따라 거래검증 및 블록생성이 이루어짐
활용사례	Bitcoin, Ethereum 등	나스닥 링크(Linq) 등	R3CEV, Tendermint 등

자료: 금융보안원, 한화투자증권

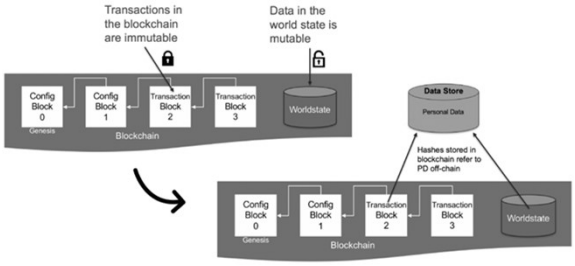
암호화폐가 불필요한 Private Blockchain을 단순한 분산형 데이터베이스로 보는 시각도 있음

james@jslab.kr

I. 개요

❖ **Off-chain : Why we have to consider off-chain model**

- **온체인은 네트워크에서 발생하는 모든 전송 내역을 블록체인 상에서 거래**
- **오프체인은 블록체인 밖에서 거래 내역을 기록하는 방식으로, 속도와 확장성 문제를 해결 가능하나 프라이버시에 취약한 블록체인 기술을 보완하기 위해 개인정보는 특정 서버에 두고, 요약 정보만 블록체인에 올리는 혼합 기법(on-chain/off-chain)을 쓰기도 함.** (예: 블록체인으로 진료 기록을 공유할 때 진료 기록원본은 내 휴대폰에 두고, 요약 정보만 블록체인에 올려 속도가 빨라지고 위변조 방지에는 효과가 있지만 블록체인의 투명성과 가용성 훼손의 단점)

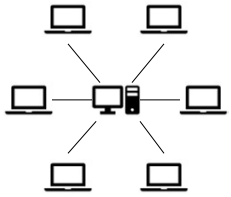


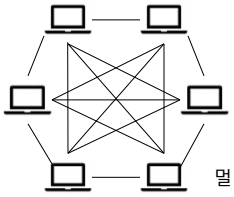
james@jslab.kr

I. 개요

❖ P2P(Peer-to-Peer) 네트워크 아키텍처

- 피어들은 네트워크를 유지하기 위한 컴퓨팅 파워와 스토리지를 기부함
- P2P 네트워크는 일반적으로 중앙집중 네트워크보다 공격(single point of attack)에 대해 안전하게 여겨짐
- Permissionless P2P 시스템은 온라인을 위해 피어 수에 대한 제한이 필요하지 않으나 일반적으로 느림
- Permissioned P2P 네트워크는 통신 회선의 QoS와 가용성(Uptime)의 일정 수준이 필요함





멀티캐스트 사용 가능

james@jslab.kr

I. 개요

❖ 글로벌 동향:

- 뉴욕에서 비트코인 규제를 위한 BitLicense도입하여 관련 사업자는 이를 취득
- 중국은 정부가 적극적으로 인프라 구축등 기술 선도
- 영국은 정부에서 'Beyond Blockchain 전략' 제시
- 일본은 블록체인 기술은 물론 암호 화폐 거래에도 개방적
- 에스토니아 디지털 시민권 (e-Estonia, e-Residents)
- 스위스 크립토 밸리(Crypto Valley)로 규제의 샌드박스 제도는 ICO 증가

순위	국가	특허 건수
01	중국	790
02	미국	762
03	한국	161
04	호주	136
05	캐나다	67
06	인도	67
07	영국	36
08	싱가포르	28
09	일본	12
10	필리핀	3

2018년 국가별 블록체인 특허 출원 건수

출처 : 더비체인(http://www.thebchain.co.kr)

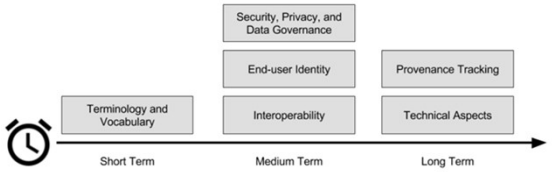
james@jslab.kr

I. 개요

❖ **표준 (Standards)**

- **ISO TC 307:** 블록체인과 분산 원장 기술을 위한 ISO 국제 표준기구가 2016년 설립되었고 미래의 표준을 위한 영역을 정의 중임(Clare Naden, 2017)
  - ✓ Terminology and concepts (2020)
  - ✓ Overview of privacy and personally identifiable information protection (no date)
  - ✓ Security risks and vulnerabilities (no date)
  - ✓ Overview of identity (no date)
  - ✓ Reference architecture (2021)
  - ✓ Taxonomy and Ontology (unclear)
  - ✓ Legally binding smart contracts (2021)
  - ✓ Overview of and interactions between smart contracts in blockchain and DLT systems (no date)

STANDARDS ROLE IN SUPPORTING BLOCKCHAIN/DLT



https://www.iso.org/news/Ref2188.htm https://www.ledgerinsights.com/iso-blockchain-standards/

james@jslab.kr

I. 개요

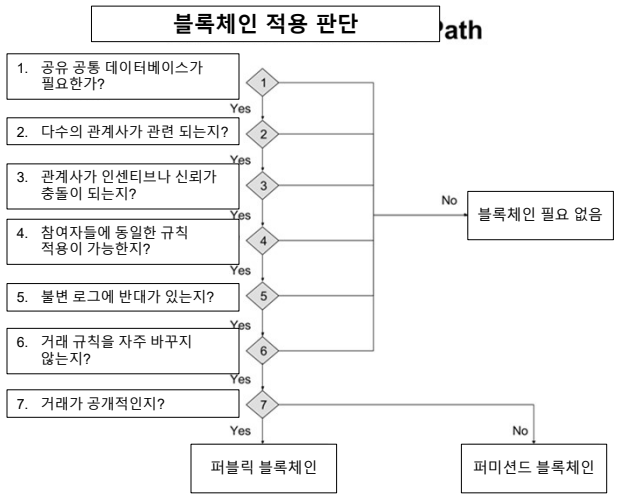
❖ **분산원장 기술 사용시 어려움**

- **표준의 부족**
- **규제**
- **분산 원장 기술의 지식과 경험 부족**
- **신 기술에 대한 저항**

james@jslab.kr

I. 개요

❖ 블록체인 기술의 비즈니스 적용 판단

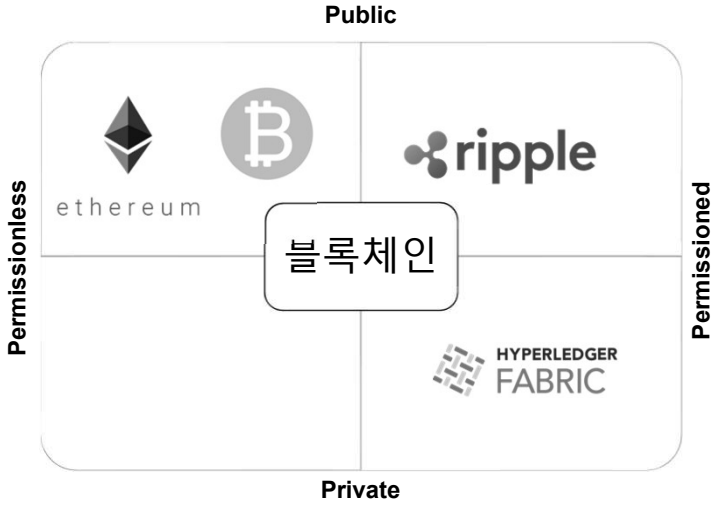


https://imgur.com/r/CryptoCurrency/wl7YI

37

I. 개요

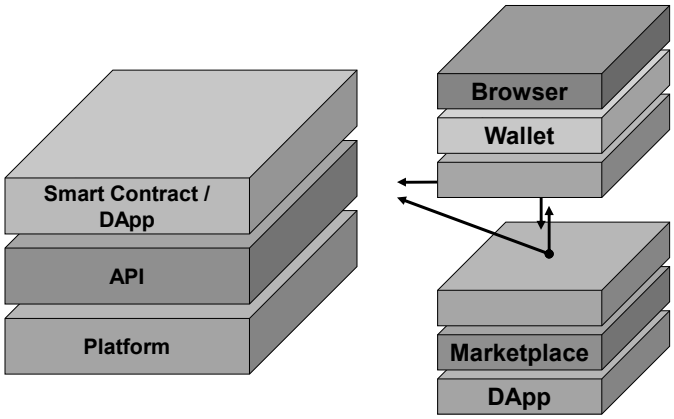
❖ BLOCKCHAIN ≠ BITCOIN



38

I. 개요

❖ **Applications**



39

I. 개요

❖ **블록체인 앱 (Blockchain Applications)**

- **블록체인은 디지털 인프라의 새로운 형태로 블록체인 앱은 블록체인 상에서 제공하는 게이트웨이를 사용하여 정보를 접속**
- **클라이언트/사용자들은 앱을 통해 블록체인과 상호 작용**
- **비트코인을 연결하는 단순 지갑(Wallet) 앱 부터 디지털 표식을 참조하거나 블록체인 상에서 금융거래를 수행하는 복잡한 앱을 포함한다.** (e.g. UPort, KYC-Chain, Netki, etc.)

free software projects for Bitcoin

•Armory - A wallet with enhanced security features, written in C++.
•BFGMiner - A modular miner, written in C.
•Bitcoin Wallet - A SPV wallet for Android, written in Java.
•bitcoinj - A library for SPV wallets, written in Java.
•btcd - A full node, written in Go.
•btcwallet - A hierarchical deterministic wallet daemon, written in Go.
•ckpool - A fast mining pool server application, written in C.
•Electrum - A fast server-trusting wallet, written in Python.
•Eloipool - A fast mining pool server application, written in Python.
•Haskoin - An implementation of the Bitcoin protocol, written in Haskell.
•Libbitcoin - A cross-platform development toolkit, written in C++.
•Libbitcoin Server - A full node and query server, built on libbitcoin.
•Libbitcoin Explorer - A command line tool, built on libbitcoin.
•Libblkmaker - A client library for the getblocktemplate mining protocol, written in C.
•NBitcoin - A cross-platform library, written in C#.
•picocoin - A tiny library with lightweight client and utilities, written in C.
•python-bitcoinlib - A library for structures and protocols, written in Python.
•Python Blkmaker - A client library for the getblocktemplate mining protocol, written in Python

40

I. 개요

❖ 요약

- 특징
  - 임의 조작 어려움
  - 해시 함수를 통한 암호화
  - 데이터 분산 저장
  - 객관적이고 투명한 관리
  - 중앙시스템이 필요 없는 탈중앙화 시스템구조(확장성, 비용절감)
  - 익명성 제공
- 활용
  - 암호화폐 : 분산성, 투명성, 보안성
  - 주식거래 : 효율성
  - 스마트 계약 : 위/변조의 어려움, 문서 보장
  - IoT : 제품 스스로 주문활용
  - 5G : SP 간 로밍 사용 과금

탈중앙성(De-centralization)
투명성(Transparency)
불변성(Immutability)
가용성(Availability)

james@jslab.kr

41

I. 개요
II. 퍼블릭 블록체인 (Public Blockchain)
III. 스마트 계약 (Smart Contract)
IV. 프라이빗 블록체인 (Private Blockchain)
V. 컨소시엄 블록체인 (Consortium Blockchain)
VI. 하이퍼레저 (Hyperledger)
VII. 컨센서스 (Consensus)
VIII. 사례 (Use Cases)
❖ 실습 (별도 교재)

james@jslab.kr

JS Lab

42

II. 퍼블릭 블록체인 (Public Blockchain)

❖ **비트코인 Public Blockchain (Permissionless)의 기술 특징:**

- **새로운 지불 시스템:** 합의된 결제 네트워크이자 디지털 신종 화폐
- **공개키 암호화 기술:** 익명성 기반의 암호화 제공 (Anonymous PKI-based)
- **P2P 분산 네트워크:** 자발적 참여로 거래 확증
- **번복 불가 시스템:** 가짜 지불 방지로 판매자 보호
- **타임 스템프:** 거래시각을 입증하여 이중지불 방지
- **통화공급량 제한:** 통화 공급의 안정성
- **오픈소스:** 시스템 운영 소스공개로 80%이상 동의로 누구나 수정 가능

❖ **이더리움 Public Blockchain (Permissionless)의 기술 특징:**

- **블록체인 앱 플랫폼**(오픈소스로 공개한 Blockchain App Platform)
- **스마트 계약 가능:** 거래 내역과 함께 계약 등의 추가정보 기록 가능하여 입력한 조건이 만족하면 계약을 실행 하도록 코딩
- **EVM(Ethereum Virtual Machine):** 스마트계약 생성시 동작하여 스마트 계약의 검증과 실행을 블록체인에 기록

** **Public Blockchain(Permissionless)**은 주어진 네트워크 인프라 환경에서 기술 구현
** **Private Blockchain 과 Consortium Blockchain(Permissioned)**에서는 네트워크 인프라 구성 고려

james@jslab.kr

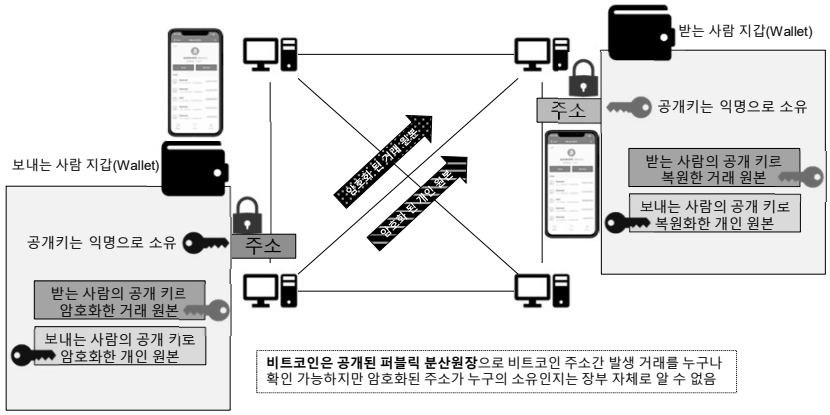
https://github.com/bitcoin/bitcoin

43

II. 퍼블릭 블록체인 (Public Blockchain)

❖ **가상화폐 거래 (PKI 기반):**

- **Wallet(지갑)**
- **개인키/공개키 생성** (공개키는 익명으로 소유)
- **주소생성**



비트코인은 공개된 퍼블릭 분산원장으로 비트코인 주소간 발생 거래를 누구나 확인 가능하지만 암호화된 주소가 누구의 소유인지는 장부 자체로 알 수 없음

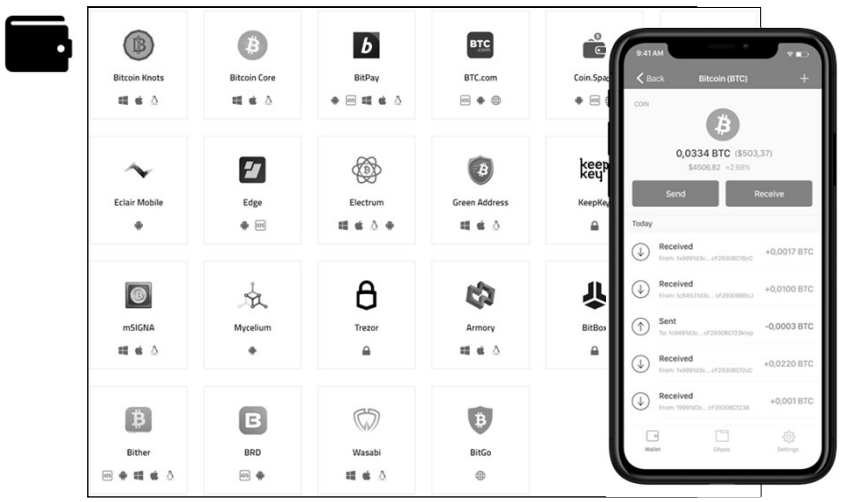
james@jslab.kr

https://bitcoin.org/en/choose-your-wallet

44

II. 퍼블릭 블록체인 (Public Blockchain)

❖ **비트코인 지갑(Bitcoin Wallet):** 저장하고 보관하는 것은 비트코인이 아닌 접근 권한을 부여하는 개인 키 (bitcoin.org는 코드를 공개한 지갑만 표시)



https://bitcoin.org/en/choose-your-wallet

II. 퍼블릭 블록체인 (Public Blockchain)

❖ **채굴(mining):**

- **입출금 내역을 검증하고 이를 포함하는 신규블록을 생성하는 작업**
- **채굴자에게는 보상으로 비트코인을 발행**
- **채굴자는 거래내역을 시스템 요구 해시함수 목표값을 찾기 위해 경쟁**

❖ **작업증명(Proof of Work):**

- **해싱을 통해 목표 값을 찾는 작업**
- **전(前) 블록의 해시값, 미승인 거래기록, 논스(nonse)에 임의 값을 입력하여 새 블록의 해시값을 계산해 목표 값과 비교하는 과정을 반복**

❖ **블록 생성:**

- **목표 값을 찾는데 성공한 참여자가 블록 발행, 이를 네트워크에 전파**
- **블록에는 전(前) 블록의 해시값, 현재 블록의 타임스탬프, 해시 목표값(난이도), 논스(Nonse)값, 현재 블록에 기록된 거래내역을 포함하는 해시 값을 기록**

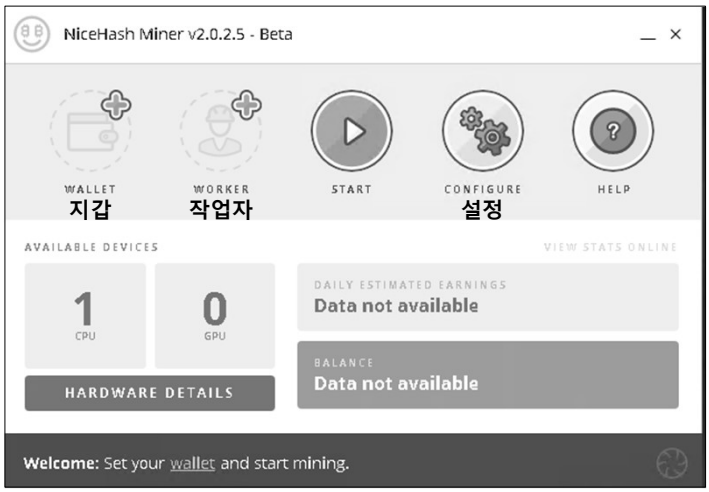
- **채굴**은 해싱작업을 통해 임의로 생성한 값을 목표값과 대조해보는 작업이기 때문에 컴퓨팅 파워를 많이 투입 할 수록 유리하고 채굴 초기에는 일반적인 개인 컴퓨터에서도 가능 했으나 거래와 채굴노드가 증가 할 수록 채굴의 난이도가 높아져 전문 장비들을 활용
- 계산(채굴) 중에 다른 노드가 생성한 블록을 감지하면 다음 블록을 만드는 작업으로 전환

II. 퍼블릭 블록체인 (Public Blockchain)

❖ **채굴 소프트웨어** (예)

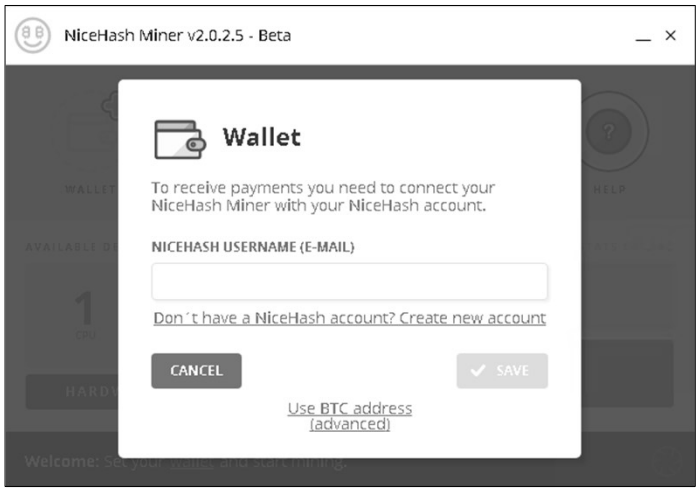
II. 퍼블릭 블록체인 (Public Blockchain)

❖ **채굴 소프트웨어 화면** (예: NiceHash Miner)



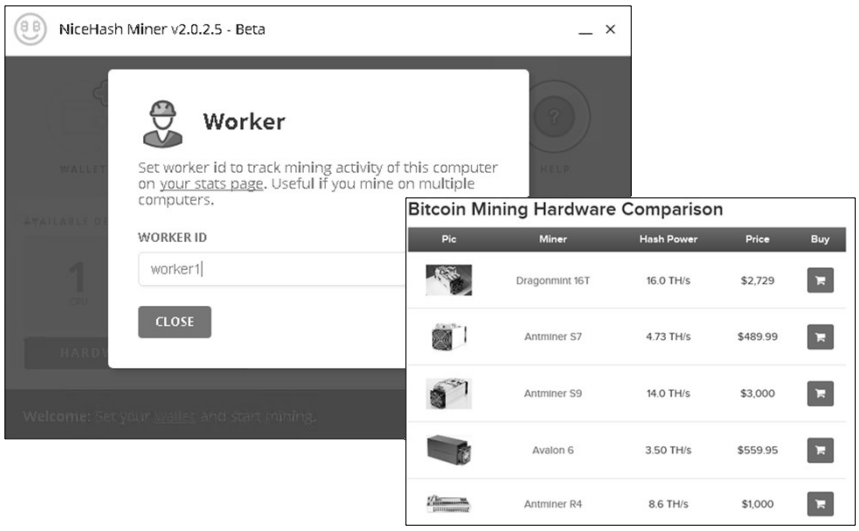
II. 퍼블릭 블록체인 (Public Blockchain)

❖ 채굴자의 지갑



II. 퍼블릭 블록체인 (Public Blockchain)

❖ **채굴자의 워커 (예)** (다수의 컴퓨터 사용시 편리)



II. 퍼블릭 블록체인 (Public Blockchain)

❖ **비트코인 채굴(mining) 보상:**

- **비트코인은 총 공급량 2100만개로 제한**
- **블록당 채굴 비트코인은 21만번째 블록생성마다 절반으로 감소하도록 설계되어 약 4년 마다 절반으로 감소** (2009년 블록 발행 보상은 50 비트코인이었으며, 2013년에는 25비트코인, 2016년은 12.5 비트코인으로 감소)
- **현재 약 1700만개 이상 채굴 되었고 채굴 종료는 2140년으로 예상**

❖ **비트코인 분산원장 용량:**

- **2009년 최초 블록 생성 후 약 10분마다 1 MB의 새로운 블록이 생성**
- **블록당 약 2000건의 거래를 저장**
- **2018년 현재 약 180 GB 정도로 개인 PC에서도 저장 가능**

높이	시각	릴레이된 곳	해시	데이터량(kB)
578952 (주 체인)	2019-06-02 15:15:24	SlushPool	000000000000000000219fb3267c30baa9195432f21aa3dbf02fe213d54b80ad	1,240.35
578951 (주 체인)	2019-06-02 15:04:20	ViaBTC	00000000000000000009dfa560e9a463b4043765a86889173e73c4c15b57dac0	1,519.77
578950 (주 체인)	2019-06-02 14:28:38	ViaBTC	00000000000000000001749aa8cd50814d4225bc9748befddf1825f3f6ec7580e	1,227.72
578949 (주 체인)	2019-06-02 14:12:49	BTC.com	00000000000000000016b04ccc4f12eef8d95a8b9b75e5f57f0798034f5f8ace1	1,170.63

https://www.blockchain.com/btc/blocks

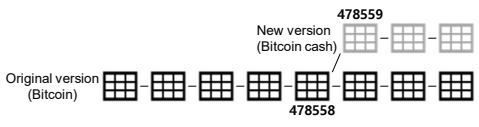
james@jslab.kr

51

II. 퍼블릭 블록체인 (Public Blockchain)

❖ **소프트포크와 하드포크:**

- **블록체인은 동시에 새로운 블록이 발생가능 하며 긴 체인을 참으로 인정**
- **비트코인은 누구나 소프트웨어 수정안을 제시 가능하며 참여자들이 동의하면 새로운 버전의 소프트웨어를 다운로드 받아 실행**
- **이전 버전으로 생성한 블록과 호환하는 방법은 소프트포크(Soft fork), 호환되지 않는 변경을 하드포크(Hard fork)라 함**
- **2017년 8월 1일 비트코인에서 비트코인캐시(Bitcoin Cash)가 478558번 블록에서 분기되는 하드포크 발생** (블록 사이즈와 처리속도 개선 의견에 대한 합의가 되지 않아 갈라짐)

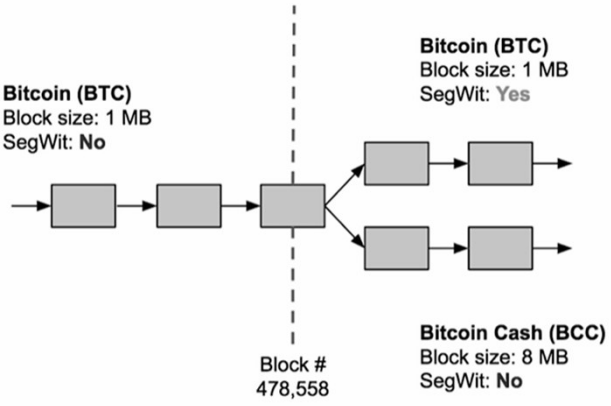


james@jslab.kr

52

II. 퍼블릭 블록체인 (Public Blockchain)

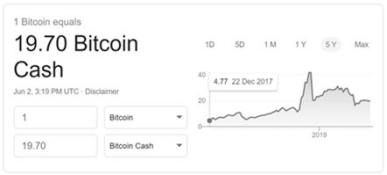
❖ 세그위트(Segregated Witness)를 활성화



II. 퍼블릭 블록체인 (Public Blockchain)

❖ 비트코인(Bitcoin)과 비트코인 캐시(Bitcoin Cash):

- 2017년 7월 20일, 비트코인 채굴자들은 97%가 비트코인 개선 제안(Bitcoin Improvement Proposal, BIP) 91에 호응을 보였다. 비트메인 워런티의 엔지니어 제임스 힐라드가 낸 이 제안은 세그위트(Segregated Witness)를 활성화하기 위한 것이었다.
- 8월 1일에 비트코인 캐시를 하드 포크로 구현할 것이라고 발표하였다. 해당 시점에서 비트코인 화폐의 거래 역사는 계승하지만 그 이후의 모든 거래는 분리되었다. 블록 478558은 마지막 공용 블록이었으로 최초의 비트코인 캐시 블록은 478559가 된다. 비트코인 캐시 암호화폐 지갑은 2017년 8월 1일 13:20 UTC 이후로 BTC 블록과 BTC 거래를 거부하기 시작했는데, 그 이유는 포크 시작을 위한 타이머가 사용되었기 때문이다. 블록 크기는 8 MB로 상승되도록 구현되어 있다.



https://ko.wikipedia.org/wiki/%EB%B9%84%ED%8A%B8%EC%BD%94%EC%9D%B8

II. 퍼블릭 블록체인 (Public Blockchain)

❖ **이더리움(Ethereum)과 이더리움 클래식:** 이더리움은 2015년 7월 30일 비탈릭 부테린(Vitalik Buterin)이 개발. 비탈릭 부테린은 가상화폐인 비트코인에 사용된 핵심 기술인 블록체인에 화폐 거래 기록뿐 아니라 계약서 등의 추가 정보를 기록할 수 있다는 점에 착안하여, 이 플랫폼을 이용하여 SNS, 이메일, 전자투표 등 다양한 정보를 기록하는 시스템을 창안.

- **2016년 6월** The DAO에 대한 해킹 사건 발생후 기존 이더리움에 대한 하드포크(Hard Fork)를 진행하여 이더리움이 두 가지 버전으로 분리. 구 버전을 이더리움 클래식(Ethereum Classic, ETC)이라고 부르고, 신 버전을 이더리움이라고 함
- **2019년 3월** 728만 번째 블록을 기준으로 콘스탄티노플(Constantinople) 하드 포크가 이루어졌다. 이를 통해서 이더리움 가상머신의 성능을 높이고, 인플레이션에 대비하기 위하여 이더리움 채굴 보상을 3개에서 2개로 줄였다.

이더리움 Roadmap

Version	Code name	Release date
0	Olympic	May, 2015
1	Frontier	30 July 2015
2	Homestead	14 March 2016
3	Metropolis (vByzantium)	16 October 2017
3.5	Metropolis (vConstantinople)	28 February 2019
4	Serenity	TBA

Old version **Latest version** Future release

이더리움 클래식

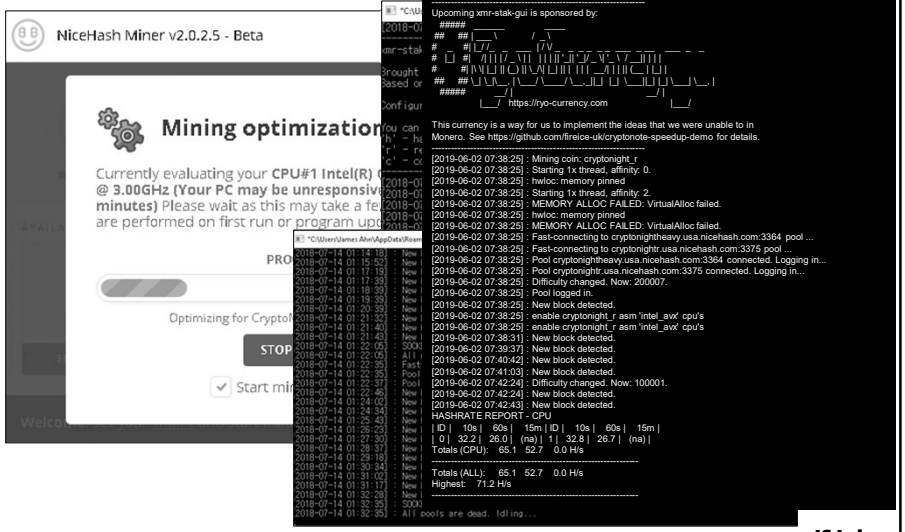
Release Date	Code name	Hard Forks and Soft Forks
30 July 2015	Frontier	The release of the Ethereum Genesis block.
14 March 2016	Homestead	The 2nd major release of the Ethereum platform, which introduced EIP-2, EIP-7, and EIP-8.
25 October 2016	GasReprice	First fork after being renamed "Ethereum Classic". Repriced some operations to prevent DoS attacks affecting both Ethereum and Ethereum Classic networks. Introduced ECIP-1050.
14 January 2017	Die Hard	Delayed the difficulty bomb which was originally intended to force the network to move from proof-of-work to proof-of-stake and added replay protection to prevent transactions on the Ethereum network being accepted on the Ethereum Classic chain. Introduced ECIP-1010 and EIP-155.
11 December 2017	Monetary policy change	Change unlimited token emission to a fixed-cap monetary policy similar to bitcoin with a hard cap of around 210 Million.

https://en.wikipedia.org/wiki/Ethereum_Classic

55

II. 퍼블릭 블록체인 (Public Blockchain)

❖ **채굴 정보 확인**



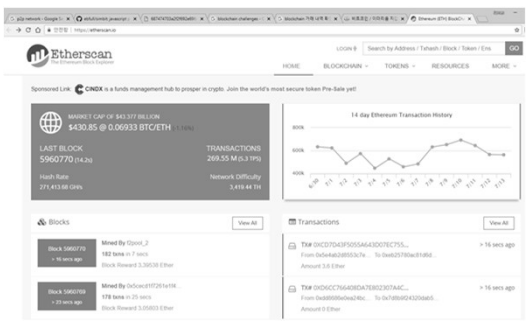
56

II. 퍼블릭 블록체인 (Public Blockchain)

❖ 컨센서스 **PoW**(Proof of Works)

- **Hash based**
- **분산 p2p 기반 Consensus Protocol 사용하여 sybil node 감지**
- **긴 블록 선호 정책**
- **구성원에게 동기 부여 (채굴)**

❖ 체인(Chain) 공개:



https://etherscan.io/

II. 퍼블릭 블록체인 (Public Blockchain)

❖ **Ethereum의 DApps (Decentralized Applications)**

- **"Ethereum은 누구나 블록체인 기술상에서 분산앱을 사용하게하는 공개 블록체인 플랫폼이다."**
- **다양한 인터페이스를 통해 P2P 로 상호 접속 (social, financial, gaming, etc.)**
- **Ethereum 블록체인 플랫폼은 스마트 거래 등의 기능을 스크립트화 하여 네트워크 내의 노드들에서 구동**
- **비트코인과 달리 거래의 추적은 물론 프로그램화 할 수 있음.**
- **기술적으로 Ethereum은 암호화폐인 이더(ether)로 기존의 인지를 재구축하는 가상머신으로 표현**
- **모든 앱사용자와 타협(compromise)한 앱 코드의 임의 변조는 불가능(이를 위한 노드들이 능동적으로 상호작용)**

II. 퍼블릭 블록체인 (Public Blockchain)

❖ **블록체인 속성 고려 인프라: 이더리움(예)**

- **노출된 외부 계약 호출:** 악성 코드를 실행하고 제어 흐름을 변경할 수 있는 외부 계약 호출에 주의
- **공개된 퍼블릭 함수:** 나의 퍼블릭 함수는 공개되어 있으며 악의적으로 또는 순서에 관계 없이 호출 가능
- **스마트 계약 정보 공개:** 스마트 계약의 개인 정보도 공개
- **비용:** 가스 비용과 가스제한(gas limit)을 고려
- **Time Stamp:** 블록 체인의 타임 스탬프는 부정확하며, 채굴자들이 트랜잭션 실행 시간 조작 가능
- **무작위성:** 대부분의 랜덤 숫자 생성 방식은 블록체인상에서 조작 가능

james@jslab.kr

https://solidity.readthedocs.io/en/latest/security-considerations.html

II. 퍼블릭 블록체인 (Public Blockchain)

❖ **DApp의 버그와 취약성에 적절히 대응 가능한 기능 고려:**

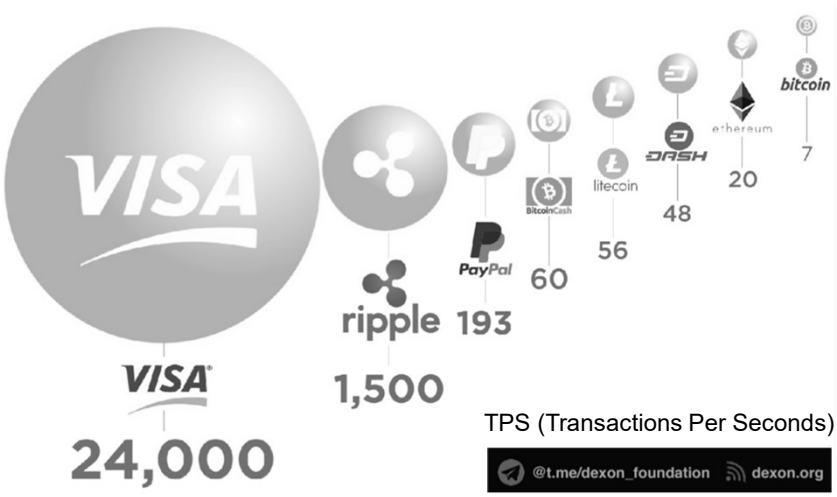
- **적용 영역:** 시스템에서 분산이 필요한 부분에만 블록 체인 적용
- **회로 차단기:** 문제가 발생하면 계약을 일시 중지
- **비용 관리:** 위험관리 (요금 제한, 최대 사용량)
- **업그레이드 고려:** 버그수정과 기능개선을 위한 효과적인 업그레이드 경로 제공

금융 샌드박스는 관련 규제를 최장 4년간 풀어주고 마음껏 영업하도록 하는 것으로, 2019년 금융혁신지원특별법에 따라 도입됐으며 문제 발생시 조치와 피해 보상을 위한 구체적 대안 제시 필요

james@jslab.kr

II. 퍼블릭 블록체인 (Public Blockchain)

❖ 처리속도



61

II. 퍼블릭 블록체인 (Public Blockchain)

❖ 처리속도 – TPS for Bitcoin

- Average of 546 bytes per transaction.
- Current blocksize is 1 MiB.
- Expected time to next block is 10 min.

Therefore we can compute the sustained maximum transaction volume in tps:

$$\frac{1\ MiB}{1\ block} \times \frac{1\ txn}{546\ bytes} \times \frac{1\ block}{10\ min} \approx 3.2\ tps$$

62

II. 퍼블릭 블록체인 (Public Blockchain)

❖ 요약

V · T · E	Cryptocurrencies	[hide]
Technology	Blockchain · Cryptocurrency tumbler · Cryptocurrency exchange · Cryptocurrency wallet · Cryptographic hash function · Distributed ledger · Fork · Lightning Network · Smart contract	
Consensus mechanisms	Proof-of-authority · Proof-of-space · Proof-of-stake · Proof-of-work	
Proof-of-work currencies	SHA-256-based	Bitcoin · Bitcoin Cash · Counterparty · MazaCoin · Namecoin · NeuCoin · Nxt · Peercoin · Titcoin
	Ethash-based	Ethereum · **Ethereum Classic**
	Scrypt-based	Auroracoin · Bitconnect · Bitcoin Gold · Coinye · Dogecoin · Gridcoin · Litecoin · PotCoin
	Equihash-based	Zcash · Zcoin
	CryptoNote-based	Monero
	X11-based	Dash · Petro
	Lyra2-based	Taler
	Other	Primecoin · Verge · Vertcoin
Proof-of-stake currencies	EOS.IO · Steem	
ERC-20 tokens	Augur · Aventus · Basic Attention Token · Centra · Kin · KodakCoin · Minds · Power Ledger	
Other currencies	BitShares · Filecoin · NEM · NEO · Ripple · Stellar · Tether	
Related topics	Airdrop · BitLicense · Blockchain game · Complementary currency · Crypto-anarchism · Cryptocurrency bubble (2018 cryptocurrency crash) · Digital currency · Double-spending · Initial coin offering · Initiative Q · List of cryptocurrencies · Stablecoin · Token money · Virtual currency	
Category · Commons · List		

https://en.wikipedia.org/wiki/Ethereum_Classic

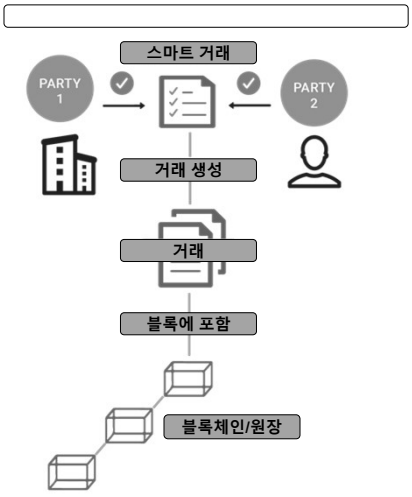
I. 개요
II. 퍼블릭 블록체인 (Public Blockchain)
III. 스마트 계약 (Smart Contract)
IV. 프라이빗 블록체인 (Private Blockchain)
V. 컨소시엄 블록체인 (Consortium Blockchain)
VI. 하이퍼레저 (Hyperledger)
VII. 컨센서스 (Consensus)
VIII. 사례 (Use Cases)
❖ 실습 (별도 교재)

III. 스마트 계약 (Smart Contract)

❖ **Smart Contracts**

- **스마트 계약(Smart contract)은 거래가 지정한 상태를 만날 때 정한 일을 수행하는 컴퓨터 프로그램**
- **스마트 계약은 원장을 갱신하는 거래의 언어를 제공**
- **어떠한 가치를 보내거나 교환하는 것을 촉진** (주식, 돈, 컨텐츠, 자산)

Smart contract = automated process



james@jslab.kr

65

III. 스마트 계약 (Smart Contract)

❖ **Smart Contract DApp (예) CryptoKitties**

- **2017년 11월 출시된 크립토키티는 블록체인을 이용한 게임이다. 세상에서 하나밖에 없는 고유한 유전자를 갖고 있는 고양이를 사서 수집하고 다른 종과 교배해 새로운 유전자를 지닌 종을 탄생시킬 수 있다. 블록체인 기술을 사용하므로 한 번 구매하면 게임 회사가 사라지더라도 영원히 내 것이며, 불법 복제나 위·변조도 불가능하다. 2018년 4월 기준 1만 달러 이상의 고양이가 100마리 이상 거래되었고 10만 달러 이상의 고양이도 탄생했다. 이더리움 전자 지갑을 만든 뒤 크립토키티 사이트에서 마음에 드는 고양이를 구입하고 키우면 된다.**

http://www.cryptokitties.co/

james@jslab.kr

66

III. 스마트 계약 (Smart Contract)

❖ 스마트 계약의 기존과 다른점

- 블록체인에 거래를 기록시 쉽고 단순하게 하는 많은 솔루션들을 자동화 할 수 있음

Traditional contracts	Smart contracts	
1-3 Days	Minutes	시간
Manual remittance	Automatic remittance	송금 자동화
Escrow necessary	Escrow may not be necessary	제3자 기탁금
Expensive	Fraction of the cost	비용
Physical presence (wet signature)	Virtual presence (digital signature)	디지털 서명
Lawyers necessary	Lawyers may not be necessary	변호사

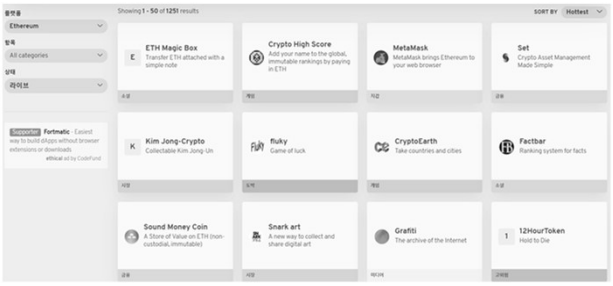
james@jslab.kr

https://www.edureka.co/blog/smart-contracts/

III. 스마트 계약 (Smart Contract)

❖ 이더리움의 스마트 계약(Ethereum Smart Contracts)

- 스마트 계약은 투자금이 필요한 회사와 투자자 사이의 협약을 암호화 할 수 있음 ①
- 스마트 계약은 이더리움 퍼블릭 블록체인에 존재하며 Ethereum Virtual Machine (EVM) 상에서 동작함 ②
- 이벤트가 발생하면 유효 기간이 있는 것과 같이 코딩된 가격을 확인하고 자동으로 스마트 거래의 비즈니스 로직을 실행 ②
- 단속자(Regulator)가 시장의 활동을 감시 할 수 있으나 특정인의 ID를 손상하지 않는 장점이 있음 ③



james@jslab.kr

https://www.stateofthedapps.com/ko/dapps/platform/ethereum?page=1&status=live

III. 스마트 계약 (Smart Contract)

❖ 스마트 계약(Smart contract):

- 이더리움은 스마트 계약을 분산 어플리케이션(Decentralized application or DApp)이라 하며 1500 여개의 DApp이 만들어짐 (토큰 발행 가능)
- 이더리움은 비트코인과 같은 결제/송금 가능한 외부소유계정(Externally Owned Account)과 스마트 계약 가능 계약계정(Contract Account)으로 구분하며, 스마트계약 구동 시 컴퓨팅 파워 사용을 위한 가스(gas)필요
- 하이퍼레저패브릭(Hyperledger Fabric)은 Chaincode로 스마트계약 수행

	이더리움	하이퍼레저 패브릭	R3 Corda
프로그램 언어	Solidity	Go, Java	Kotlin
거버넌스	참여자들의 분산 처리	체인 내에 리눅스 재단과 조직	R3와 관련 기관들
스마트 계약	법적 제한 없음	법적 제한 없음	법적 제한
컨센서스 알고리즘	PoW, PoS	PBFT, Pluggable Consensus	다수 컨센서스 알고리즘 동작 서기(Notary) 노드
확장성	확장성 이슈 있음	강조되지 않음	강조되지 않음
프라이버시	프라이버시 이슈 있음	강조되지 않음	강조되지 않음
통화	이더(Ether)	없음 (Chaincode 사용 생성 가능)	없음

69

III. 스마트 계약 (Smart Contract)

❖ 스마트 계약(Smart contract) 2018:

Characteristics	Ethereum	HyperLedger Fabric	R3 Corda
Description of the Platform	Generic Blockchain Platform	Modular Blockchain Platform	Specialized distributed ledger platform for Financial Industry
Release History	July 2015	v0.6 Sept 2016, v1.0 July 2017, **v1.3 2018**	v-m0.0 May 2016, v1.0 Oct 2017, **v3.0 Mar 2018**
Crypto Currency	Ether / Tokens (Usage, Work) via Smart Contract	None Currency and Tokens via Chaincode	None
Governance	Ethereum Developers Enterprise Ethereum Alliance	Linux Foundation IBM	R3
Consensus	Mining based on Proof of Work (POW) – All participants need to agree. Ledger Level	Selective Endorsement. Consensus can be even within a channel with select parties instead of everyone. Transaction Level	Specific understanding of Consensus. (Validity, Uniqueness) Transaction Level
Network	Permissionless, Public or Private	Permissioned, Private	Permissioned, Private
State	Account Data	Key-value Database Transaction Log, World State	Vault contains States Historic & Current State
Smart Contracts	Solidity	Chaincode (GoLang, Node.JS, Java)	Smart Contract (Kotlin, Java)
Development Languages	GoLang, C++, Python	Java, Node.JS, Python (Post 1.0)	Java, Kotlin

ARAF KARSH HAMID
Co-Founder / CTO
MetaMagic Global Inc., NJ, USA

70

III. 스마트 계약 (Smart Contract)

❖ Enterprise Blockchain 비교 (성능 참고)

	Hyperledger Fabric	Quorum	Ethereum	Ripple	R3 Corda
Ledger Type	Permissioned	Permissioned	Permissionless	Permissioned	Permissioned
Governance	Linux Foundation	JP Morgan and Ethereum developers	Ethereum developers	Ripple Labs	R3 Consortium
Industry-focus	Cross-Industry	Multi-Industry	Cross-Industry	Financial Industry	Financial Industry
Throughput	> 2000 tps	100 tps	~ 20 tps	~ 1500 tps	~170 tps
Cryptocurrency	None	None	Ether (ETH)	Ripple (XRP)	None
Consensus Mechanism	Pluggable Mechanism	Voting Protocol	Proof of Work (PoW)	Probabilistic Voting Protocol	Pluggable Mechanism
Smart Contract	✓	✓	✓	×	✓
Smart Contract Language	NodeJS or Golang or Java	Solidity	Solidity	-	Java or Kotlin
Application Type	Wide-ranging	Broad-ranging	Wide-ranging	Suited for financial applications	Financial applications

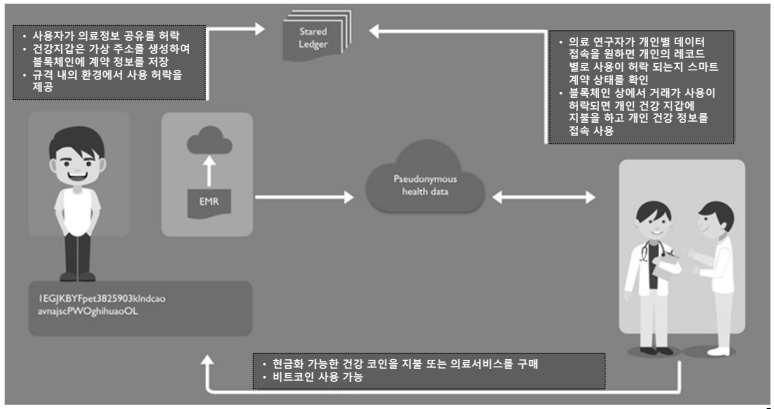
james@jslab.kr

71

III. 스마트 계약 (Smart Contract)

❖ Use Case: Healthcare Industry

- 각 환자는 유일하기 때문에 개인에 맞는 진찰을 선택하기 위해 의료 정보를 접속
- 의료 커뮤니티에서는 정보 공유가 주요 어려움
- 블록체인 기반의 스마트 계약으로 해결



https://www.edureka.co/blog/smart-contracts/ https://medibloc.org/ko/

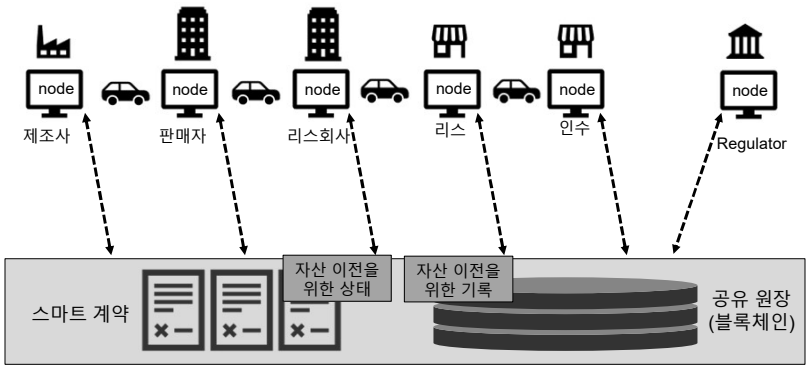
james@jslab.kr

72

III. 스마트 계약 (Smart Contract)

❖ **Use Case: 블록체인 기반 자동차 리스 비즈니스 네트워크**

- 컨소시엄이 필요하고, 멀티클라우드 기반의 인프라가 요구됨
- 블록체인 상에서 개별 원장을 사용하는 복수의 채널을 가질 수 있음

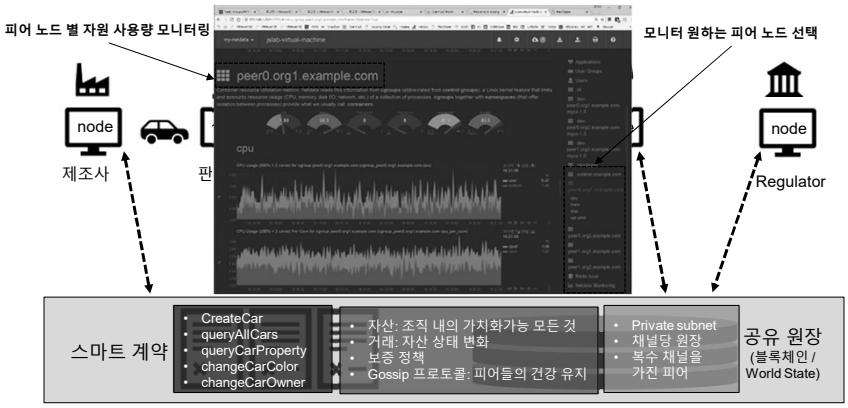


73

III. 스마트 계약 (Smart Contract)

❖ **Use Case: 블록체인 기반 자동차 리스 비즈니스 네트워크**

- 노드별 자원 사용 모니터링이 필요하며 확장 또는 이동이 필요 할 수 있다.

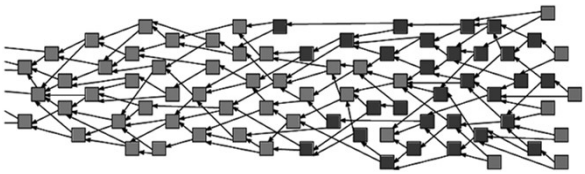


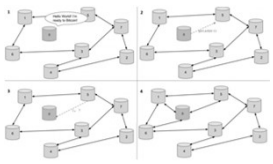
74

III. 스마트 계약 (Smart Contract)

❖ Gossip Protocol

- Manages peer discovery and channel membership, by continually identifying available member peers, and eventually detecting peers that have gone offline.
- Disseminates ledger data across all peers on a channel. Any peer with data that is out of sync with the rest of the channel identifies the missing blocks and syncs itself by copying the correct data. A state reconciliation process synchronizes world state across peers on each channel. Each peer continually pulls blocks from other peers on the channel, in order to repair its own state if discrepancies are identified
- Bring newly connected peers up to speed by allowing peer-to-peer state transfer update of ledger data.





III. 스마트 계약 (Smart Contract)

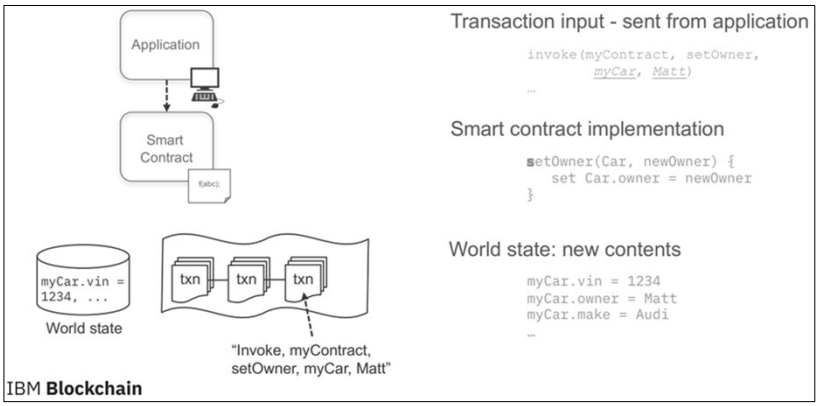
❖ 불록체인 비즈니스 클러스터링 네트워크 모니터

- 컨테이너 기반은 L4-L7 트래픽 플로우 모니터링 (예)

III. 스마트 계약 (Smart Contract)

❖ **Use Case: 불록체인 기반 자동차 리스 비즈니스 네트워크**

- **소유 변경 거래 (예)**



james@jslab.kr

III. 스마트 계약 (Smart Contract)

❖ **하이퍼레저(Hyperledger)의 스마트 계약(Smart contract):**

- **하이퍼레저 패브릭(Hyperledger Fabric)은 Chaincode로 스마트계약 수행하며 Golang과 Javascript를 사용하고, Sawtooth는 파이손과 C등 더 다양한 언어를 사용 가능**

Smart Contract Implementations in Hyperledger Frameworks

Framework	Smart Contract Technology	Smart Contract Type	Language(s) for Writing Smart Contracts
Hyperledger Burrow	Smart contract application engine	On-Chain	Native language code
Hyperledger Fabric	Chaincode	Installed	Golang (> v1.0) or Javascript (> v1.1)
Hyperledger Indy	None	None	None
Hyperledger Iroha[2]	Chaincode	On-chain	Native language code
Hyperledger Sawtooth	Transaction families	On-Chain and Installed	C++, Go, Java, JavaScript, Python, Rust, or Solidity (through Seth)

james@jslab.kr

I. **개요**

II. **퍼블릭 블록체인** (Public Blockchain)

III. **스마트 계약** (Smart Contract)

IV. **프라이빗 블록체인** (Private Blockchain)

V. **컨소시엄 블록체인** (Consortium Blockchain)

VI. **하이퍼레저** (Hyperledger)

VII. **컨센서스** (Consensus)

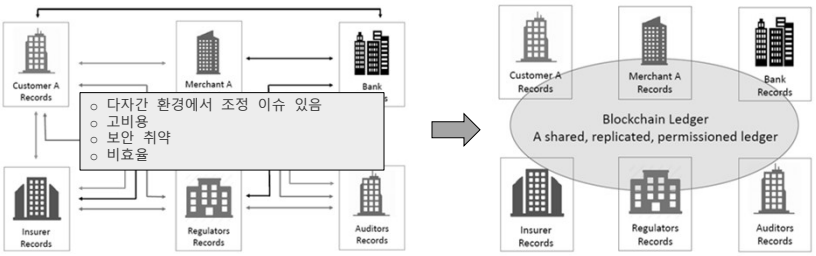
VIII. **사례** (Use Cases)

❖ **실습** (별도 교재)

IV. 프라이빗 블록체인

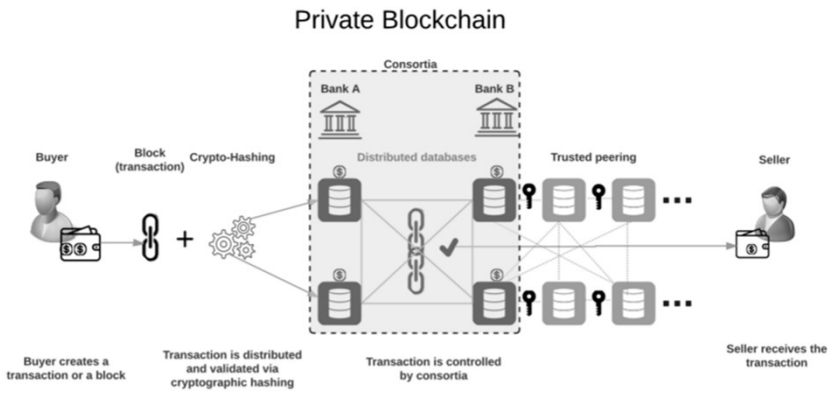
❖ **프라이빗 블록체인:** 중앙 통제 조직이나 대표자가 존재하고 허가된 참여자만 네트워크에 들어올 수 있어 참여자 간 식별이 가능하며 특화된 데이터 공유가 가능

- **거래 위임전에 참여자들 사이에 Consensus**
- **거래 변조 확인하는 Immutability**
- **자산 위치를 쉽게 추적하는 Provenance**
- **진위 확인을 한곳에서 Finality**



IV. 프라이빗 블록체인

❖ **프라이빗 블록체인:** Quorum, Hyperledger Fabric, R3 Corda



https://www.devteam.space/blog/public-vs-private-permissioned-blockchain-comparison/

I. 개요
II. 퍼블릭 블록체인 (Public Blockchain)
III. 스마트 계약 (Smart Contract)
IV. 프라이빗 블록체인 (Private Blockchain)
V. 컨소시엄 블록체인 (Consortium Blockchain)
VI. 하이퍼레저 (Hyperledger)
VII. 컨센서스 (Consensus)
VIII. 사례 (Use Cases)
❖ 실습 (별도 교재)

V. 컨소시엄 블록체인

❖ 블록체인 컨소시엄

- **Consortium Blockchain은 퍼블릭 블록체인과 프라이빗 블록체인의 중간 형태로 소유자가 모든 권한이 있는 프라이빗 블록체인과 달리 선정된 노드들이 권한을 가지는 블록체인**

컨소시엄	참여기관	특징
R3 CEV	• 미국 IT 기업 R3사 설립 • 골드만삭스등 80여개 금융 기관 • 국내 은행(국민, 신한, 하나, 우리)	• 분산 원장 플랫폼인 금융기관 계약 기록 관리 시스템(Corda) 개발
Hyperledger	• 리눅스재단 관리 • 100여개의 금융/비금융 기업 참여 • 국내기업(한국거래소, 코스콤, SDS, 한국예탁결제원)	• 오픈소스 • 범 산업용 블록체인 플랫폼
SBI 핀테크 컨소시엄	• 일본 SBI 금융 그룹 주도 • 리플, 코인 플러그등 참여	• 오픈소스 • 범 산업용 블록체인 플랫폼
차이나레져	• 중국 완샹 블록체인 랩 주도 • 중국 11개 대형 금융기관 참여	• R3와 이더리움 재단 자문
아르고	• 국내 블록체인 업체 블로코와 해외 시장을 공략하기 위한 플랫폼 개발 추진	• 클라우드 서비스 제공업체는 물론 이동통신사업자 등을 위한 플랫폼 계획

james@jslab.kr

V. 컨소시엄 블록체인

❖ Enterprise를 위한 Blockchain 분류 비교

Summary of Features of top 5 Blockchain Platforms for Enterprises

	Ethereum	Hyperledger Fabric	R3 Corda	Ripple	Quorum
Industry-focus	Cross-industry	Cross-industry	Financial Services	Financial Services	Cross-industry
Governance	Ethereum developers	Linux Foundation	R3 Consortium	Ripple Labs	Ethereum developers & JP Morgan Chase
Ledger type	Permissionless	Permissioned	Permissioned	Permissioned	Permissioned
Cryptocurrency	Ether (ETH)	None	None	Ripple (XRP)	None
% providers with experience[1]	93%	93%	60%	33%	27%
% share of engagements[2]	52%	12%	13%	4%	10%
Coin Market Cap[3]	$91.5 B (18%)	Not applicable	Not Applicable	$43.9 B (9%)	Not Applicable
Consensus algorithm	Proof of Work (PoW)	Pluggable framework	Pluggable framework	Probabilistic voting	Majority voting
Smart contract functionality	Yes	Yes	Yes	No	Yes

https://www.horsesforsources.com/top-5-blockchain-platforms_031618

james@jslab.kr

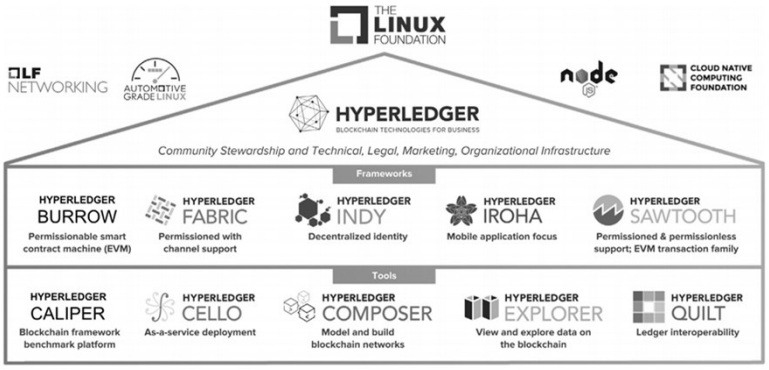
I. 개요
II. 퍼블릭 블록체인 (Public Blockchain)
III. 스마트 계약 (Smart Contract)
IV. 프라이빗 블록체인 (Private Blockchain)
V. 컨소시엄 블록체인 (Consortium Blockchain)
VI. 하이퍼레저 (Hyperledger)
VII. 컨센서스 (Consensus)
VIII. 사례 (Use Cases)
❖ 실습 (별도 교재)

james@jslab.kr

VI. 하이퍼레저(Hyperledger)

❖ **Hyperledger Umbrella Strategy**

- **Infrastructure:** Hyperledger에서 생태계를 위한 기술, 규제, 마케팅, 조직
- **Framework:** Fabric, Burrow, Indy, Iroha, Sawtooth
- **Tools:** Caliper, Cello, Composer, Explorer, Quilt



https://wiki.hyperledger.org/ https://wiki.hyperledger.org/#Hyperledger-Projects

james@jslab.kr

VI. 하이퍼레저(Hyperledger)

❖ Hyperledger

- 일반적으로 permissioned blockchain이며 네트워크 내에 허가된 참여자만 접속 가능
- 엔터프라이즈 급의 오픈소스로 분산원장프레임워크를 생성하는 것이 목표이고 비즈니스 Use Case를 지원하는 코드 기반
- 오픈소스 기반의 노력으로 향상된 산업간 블록체인 기술 생산에 노력
- 리눅스재단이 관리하며 다양한 사업과 기관들이 멤버
- 엔터프라이즈를 위해 준비된 솔루션을 알림
- 블록체인 프레임워크와 플랫폼을 만드는 소프트웨어 개발자를 위한 커뮤니티

V. 컨소시엄 블록체인

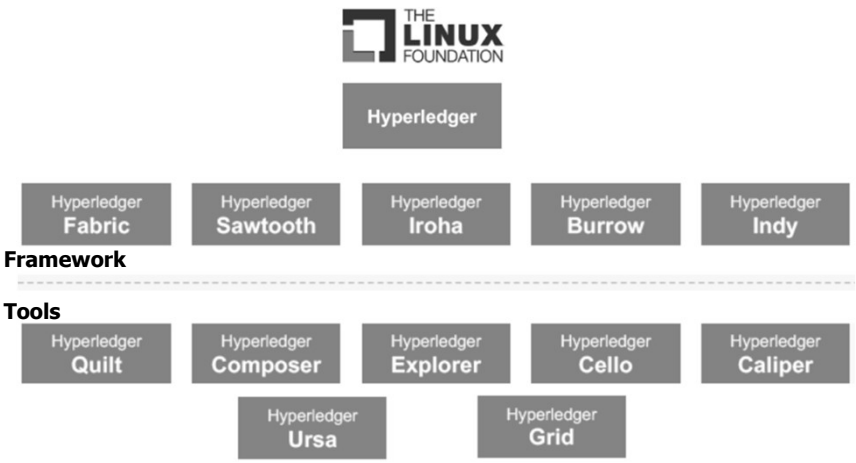
❖ Hyperledger Projects

재목	Project	Status	CII Badge	Description
Hyperledger Aries	Hyperledger Aries	INCUBATION	cii best practices not started	Infrastructure for blockchain-rooted, peer-to-peer interactions. It provides a shared, reusable, interoperable tool kit designed for initiatives and solutions focused on creating, transmitting and storing verifiable digital credentials.
Hyperledger Burrow	HYPERLEDGER BURROW	INCUBATION	cii best practices in progress 82%	Permissioned Ethereum smart-contract blockchain
Hyperledger Caliper	HYPERLEDGER CALIPER	INCUBATION	cii best practices in progress 97%	Blockchain benchmark framework which allows users to measure the performance of a specific blockchain implementation with a set of predefined use cases.
Hyperledger Cello	HYPERLEDGER CELLO	INCUBATION	cii best practices not started	Blockchain management/operation
Hyperledger Composer	HYPERLEDGER COMPOSER	INCUBATION	cii best practices passing	Development framework/tools for building Blockchain business networks
Hyperledger Explorer	HYPERLEDGER EXPLORER	INCUBATION	cii best practices in progress 86%	Blockchain Web UI
Hyperledger Fabric	HYPERLEDGER FABRIC	ACTIVE	cii best practices passing	Distributed ledger in Golang
Hyperledger Grid	HYPERLEDGER GRID	INCUBATION	cii best practices not started	Platform for building supply chain solutions that include distributed ledger components.
Hyperledger Indy	HYPERLEDGER INDY	ACTIVE	cii best practices passing	Distributed ledger purpose-built for decentralized identity
Hyperledger Iroha	HYPERLEDGER IROHA	ACTIVE	cii best practices passing	Distributed ledger in C++
Hyperledger Quilt	HYPERLEDGER QUILT	INCUBATION	cii best practices not started	An interoperability solution for blockchains, DLTs and other types of ledgers
Hyperledger Sawtooth	HYPERLEDGER SAWTOOTH	ACTIVE	cii best practices passing	Distributed ledger with Multi-Language Support
Hyperledger Ursa	HYPERLEDGER URSA	INCUBATION	cii best practices not started	A shared cryptographic library that would enable people (and projects) to avoid duplicating other cryptographic work and hopefully increase security in the process.

https://wiki.hyperledger.org/display/HYP/Projects

V. 컨소시엄 블록체인

❖ 계속 추가중인 Hyperledger Projects



https://hyperledger-fabric.readthedocs.io/en/release-1.4/

james@jslab.kr

VI. 하이퍼레저(Hyperledger)

❖ Permissioned Blockchain 장점 제공

- Permissionless blockchains은 누구나 접속 가능하여 감염 의심자가 접속 할 수 있으나 Permissioned Blockchain는 이를 방지하여 보안을 강화 가능
- Hyperledger는 참여자가 원하는 부분의 거래만 공개 가능
- Hyperledger 아키텍처는 블록체인의 모든 기능을 제공
  - ✓ data privacy
  - ✓ information sharing
  - ✓ immutability, with a full stack of security protocols
  - ✓ all for the enterprise

james@jslab.kr

VI. 하이퍼레저(Hyperledger)

❖ 주요 Hyperledger Framework Projects

- Hyperledger Fabric: supply-chain 네트워크에 사용
- Hyperledger Sawtooth: 인텔 기부, 어류 이동 추적 기반 등의 서비스체인(예)
- Hyperledger Burrow: 하이퍼레저 네트워크 내에서 Ethereum 스마트 계약 사용
- Hyperledger Iroha: 블록체인의 모바일 앱에 사용
- Hyperledger Indy: 비즈니스를 위한 ID 데이터베이스의 분산화

james@jslab.kr

https://www.edureka.co/blog/what-is-hyperledger/

VI. 하이퍼레저(Hyperledger)

❖ 하이퍼레저 패브릭 (Fabric)

- 초기 IBM이 제공한 코드를 기반으로 현재 30여 조직에서 개발 참여 중
- 2017년 7월 v1.0 출시, 2019년 5월 현재 최신 버전 v1.4
- 모바일과 웹을 위한 인프라 제공 (iOS, Android, JavaScript 지원)
- MSP(Membership Service Provider) 기반의 Identity 관리와 접근 허가
- 블록체인 원장(ledger)과 현재 상태를 나타내는 World State 보관
- 트랜잭션 블록 정하는 방법으로 'Solo', 'Kafka', SBFT 제공
- 멀티플 멤버십 서비스를 지원 Endorsing Peer(양도자), Ordering service-node(지시자) 두 가지 역할로 나눠서 수행
- Ordering Service(주문 서비스)는 네트워크의 동료와 일관되게 트랜잭션을 전달.
- 기밀 정보 공유를 위한 채널
- 블록체인 비즈니스 네트워크 안에 포함된 특정 사용자만 원장을 확인할 수 있는 권한을 채널 별로 부여
- 아파치 2.0 라이선스

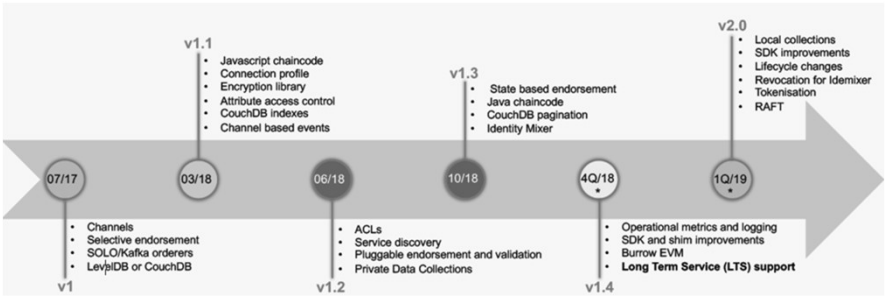
james@jslab.kr

https://bitcoinmagazine.com/articles/hyperledger-releases-sawtooth-10/

VI. 하이퍼레저(Hyperledger)

❖ 하이퍼레저 Fabric

- **Over 291 developers**
- **41 companies and 87 individuals**
- **Over 8,000 change sets**
- **v1.4. was actually released on 01/10/2019, v2.0 is subject to change**



james@jslab.kr

VI. 하이퍼레저(Hyperledger)

❖ 하이퍼레저 Fabric

- **What's New in Hyperledger Fabric v1.4.1**
  - **Hyperledger Fabric's First long term support release**
  - **Raft ordering service** - Raft Consensus: The ordering service now provides an option to use the Raft Consensus algorithm.
  - **Serviceability and operations improvements**
    - ✓ Endorser metrics
    - ✓ Orderer communication metrics
    - ✓ FRaft metrics
    - ✓ Metrics for log records
    - ✓ Kafka health check
    - ✓ CouchDB health check
  - **Improved programming model for developing applications**
  - **New tutorials**
  - **Private data enhancements**

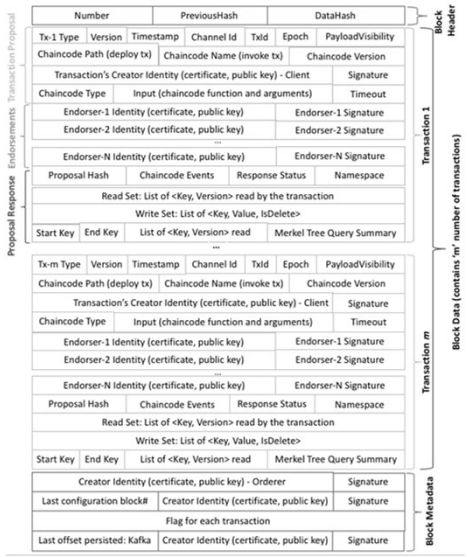
HYPERLEDGER FABRIC

https://hyperledger-fabric.readthedocs.io/en/release-1.4/whatsnew.html

james@jslab.kr

VI. 하이퍼레저(Hyperledger)

❖ **Fabric Block Structure**

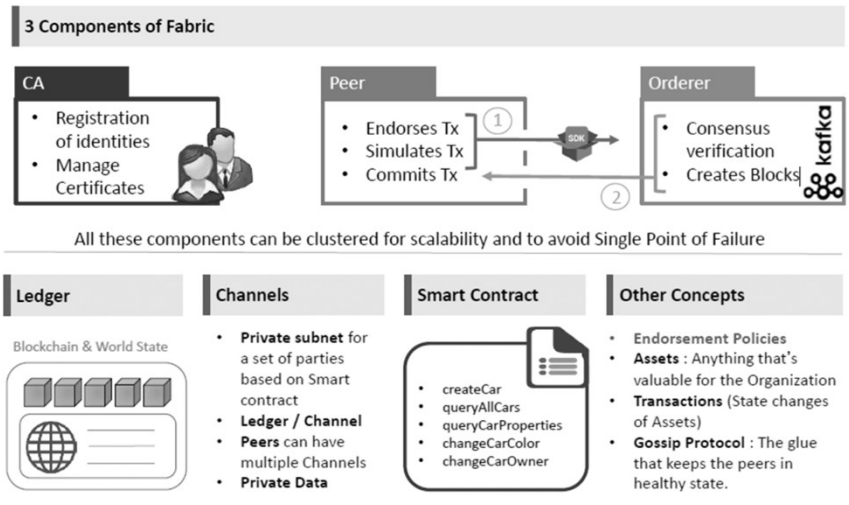


https://blockchain-fabric.blogspot.com/2017/04/hyperledger-fabric-v10-block-structure.html

95

VI. 하이퍼레저(Hyperledger)

❖ **3 components @ Hyperledger Fabric Architecture**

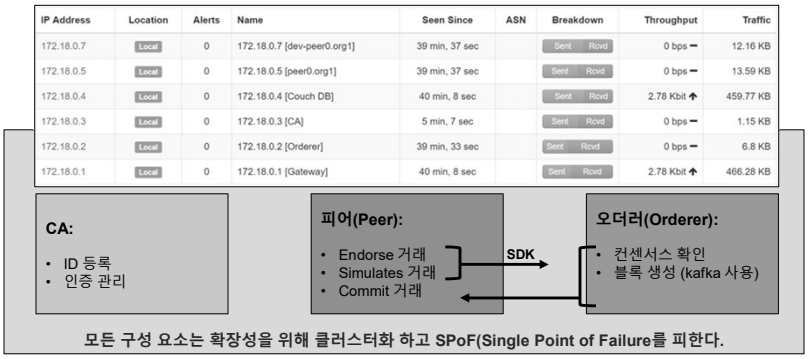


96

VI. 하이퍼레저(Hyperledger)

❖ **Fabric**

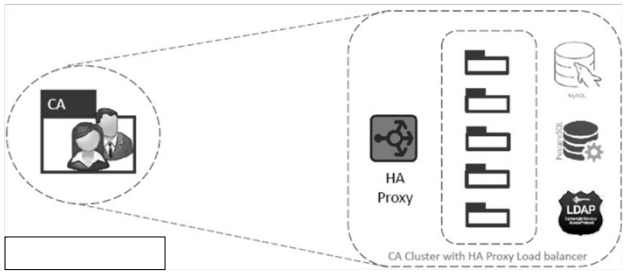
- **Architecture**
  - ✓ **CA:** ID 등록 및 인증 관리
  - ✓ **피어(Peer):** Endorse 거래, Simulates 거래, Commit 거래
  - ✓ **오더러(Orderer):** 컨센서스 확인, 블록 생성 (kafka 사용)



VI. 하이퍼레저(Hyperledger)

❖ **Fabric**

- **CA**



CA:
- ID 등록
- 인증 관리

피어(Peer):
- Endorse 거래
- Simulates 거래
- Commit 거래

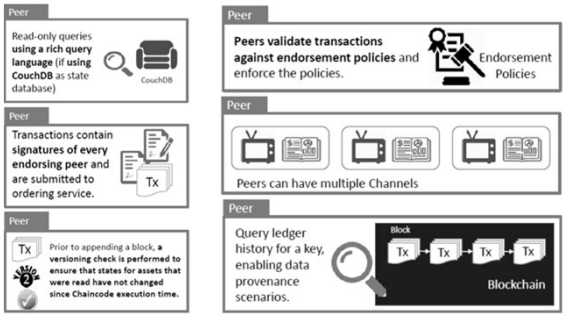
SDK

오더러(Orderer):
- 컨센서스 확인
- 블록 생성 (kafka 사용)

VI. 하이퍼레저(Hyperledger)

❖ **Fabric**

- **Peer**



CA:

ID 등록
인증 관리

피어(Peer):

- Endorse 거래
- Simulates 거래
- Commit 거래

SDK

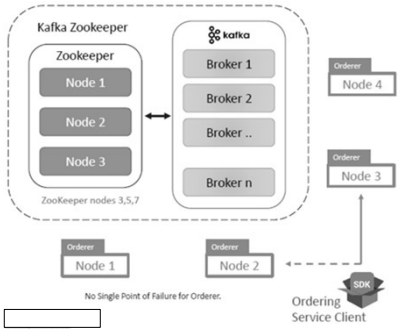
오더러(Orderer):

컨센서스 확인
블록 생성 (kafka 사용)

VI. 하이퍼레저(Hyperledger)

❖ **Fabric**

- **Orderer**



CA:

ID 등록
인증 관리

피어(Peer):

Endorse 거래
Simulates 거래
Commit 거래

SDK

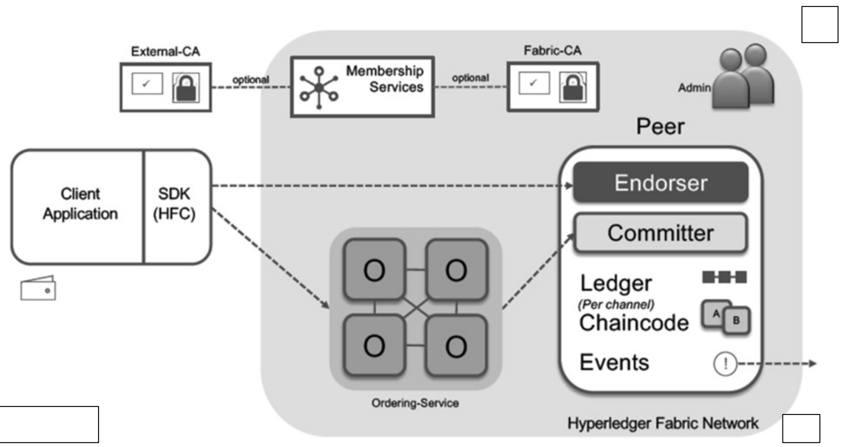
오더러(Orderer):

- 컨센서스 확인
- 블록 생성 (kafka 사용)

VI. 하이퍼레저(Hyperledger)

❖ **Fabric**

- **Hyperledger Fabric V1 architecture recap**



VI. 하이퍼레저(Hyperledger)

❖ **Fabric**

- **Hyperledger Fabric as Docker Containers**

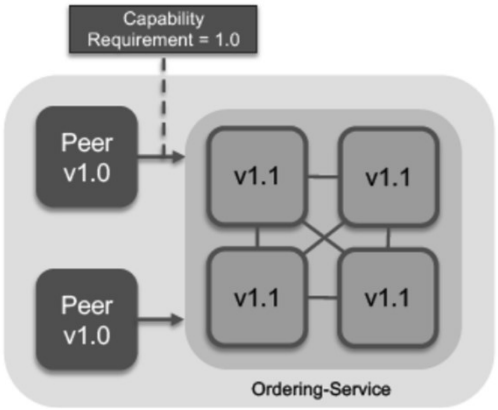
```
oot@ubuntu:/home/jslab/fabric-samples/first-network#
CONTAINER ID    IMAGE                              COMMAND                  PORTS                       NAMES
ef4f5dc66054    dev-peer1.org2.example.com-mycc-1  "chaincode -peer.add…"                               dev-peer1.org2.example.com-mycc-1.0
1570c6aec8b4    dev-peer0.org1.example.com-mycc-1  "chaincode -peer.add…"                               dev-peer0.org1.example.com-mycc-1.0
9940875947f5    dev-peer0.org2.example.com-mycc-1  "chaincode -peer.add…"                               dev-peer0.org2.example.com-mycc-1.0
69d696242805    hyperledger/fabric-tools:latest    "/bin/bash"                                          cli
47b0c1fa7128    hyperledger/fabric-peer:latest     "peer node start"        0.0.0.0:9051->9051/tcp      peer0.org2.example.com
570155a2e9be    hyperledger/fabric-peer:latest     "peer node start"        0.0.0.0:7051->7051/tcp      peer0.org1.example.com
952e9a346674    hyperledger/fabric-peer:latest     "peer node start"        0.0.0.0:8051->8051/tcp      peer1.org1.example.com
d5620602b1cd    hyperledger/fabric-peer:latest     "peer node start"        0.0.0.0:10051->10051/tcp    peer1.org2.example.com
e762ab2ad630    hyperledger/fabric-orderer:lates   "orderer"                0.0.0.0:7050->7050/tcp      orderer.example.com
root@ubuntu:/home/jslab/fabric-samples/first-network#
```

```
oot@ubuntu:/home/jslab/fabric-samples/first-network# docker ps
CONTAINER ID    IMAGE                                                                                                   COMMAND                  CREATED          STATUS          PORTS                       NAMES
ef4f5dc66054    dev-peer1.org2.example.com-mycc-1.0-26c2ef32838554aac4f7ad6f100aca865e87959c9a126e86d764c8d01f8346ab   "chaincode -peer.add…"   34 hours ago     Up 34 hours                                 dev-peer1.org2.example.com-mycc-1.0
1570c6aec8b4    dev-peer0.org1.example.com-mycc-1.0-384f11f484b9302df90b453200cfb25174305fce8f53f4e94d45ee3b6cab0ce9   "chaincode -peer.add…"   34 hours ago     Up 34 hours                                 dev-peer0.org1.example.com-mycc-1.0
9940875947f5    dev-peer0.org2.example.com-mycc-1.0-15b571b3ce849066b7ec74497da3b27e54e0df1345daff3951b94245ce09c42b   "chaincode -peer.add…"   34 hours ago     Up 34 hours                                 dev-peer0.org2.example.com-mycc-1.0
69d696242805    hyperledger/fabric-tools:latest                                                                         "/bin/bash"              34 hours ago     Up 34 hours                                 cli
47b0c1fa7128    hyperledger/fabric-peer:latest                                                                          "peer node start"        34 hours ago     Up 34 hours     0.0.0.0:9051->9051/tcp      peer0.org2.example.com
570155a2e9be    hyperledger/fabric-peer:latest                                                                          "peer node start"        34 hours ago     Up 34 hours     0.0.0.0:7051->7051/tcp      peer0.org1.example.com
952e9a346674    hyperledger/fabric-peer:latest                                                                          "peer node start"        34 hours ago     Up 34 hours     0.0.0.0:8051->8051/tcp      peer1.org1.example.com
d5620602b1cd    hyperledger/fabric-peer:latest                                                                          "peer node start"        34 hours ago     Up 34 hours     0.0.0.0:10051->10051/tcp    peer1.org2.example.com
e762ab2ad630    hyperledger/fabric-orderer:latest                                                                       "orderer"                34 hours ago     Up 34 hours     0.0.0.0:7050->7050/tcp      orderer.example.com
root@ubuntu:/home/jslab/fabric-samples/first-network#
```

VI. 하이퍼레저(Hyperledger)

❖ Fabric

- Rolling Upgrade Support
- https://jira.hyperledger.org/browse/FAB-5556

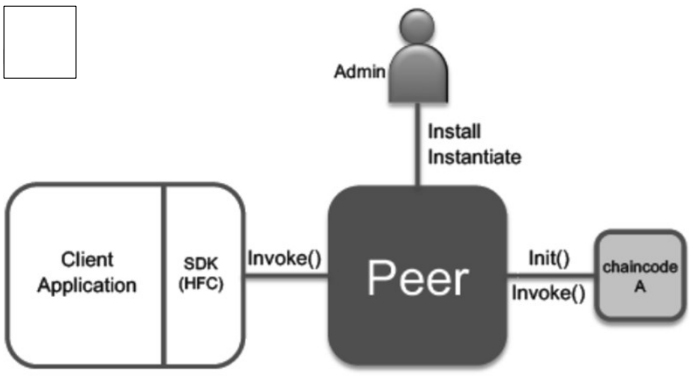


james@jslab.kr

VI. 하이퍼레저(Hyperledger)

❖ Fabric

- Chaincode (a.k.a Smart Contract)
- Chaincode contains business logic deployed to peers
- Fabric support for javascript chaincode
  - ✓ https://jira.hyperledger.org/browse/FAB-2331



james@jslab.kr

VI. 하이퍼레저(Hyperledger)

❖ Fabric

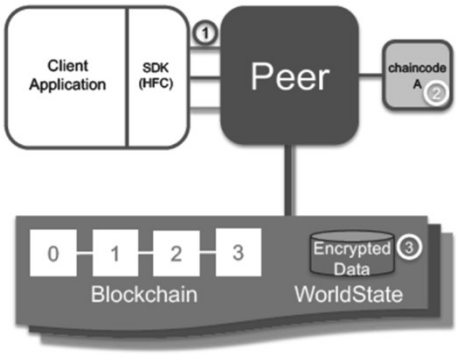
- Chaincode (a.k.a Smart Contract)

- Business logic
- Enforces rules to read or altering state
- Defining an assets
- Transaction instruction to change Asset state
- Write smart contracts in Java, Go or Node.js
- Chaincode execution results in a state change in ledger
- Governed by System chaincode's Endorsement Policy
- Each Chaincode instance in its own container

VI. 하이퍼레저(Hyperledger)

❖ Fabric

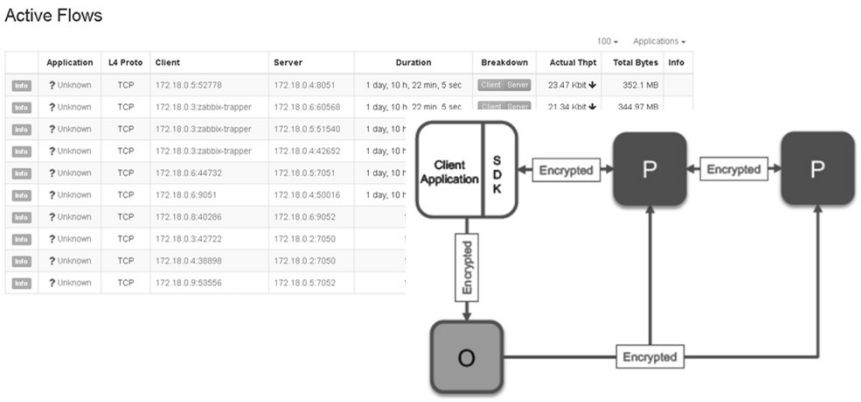
- Application Level Encryption
  1. Pass unencrypted data and keys to endorser
  2. Chaincode encrypts data to put in worldstate
  3. Encrypted data stored in worldstate



VI. 하이퍼레저(Hyperledger)

❖ **Fabric**

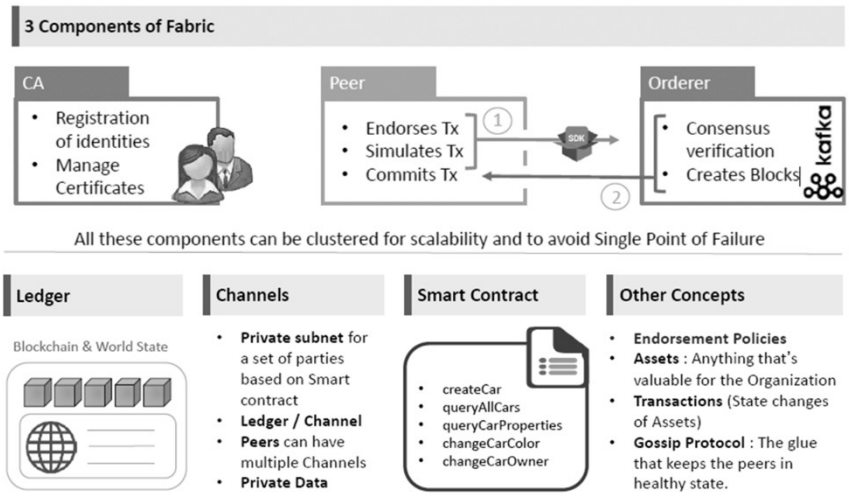
- **Transport Layer Security**
- **All communications within a Hyperledger Fabric network can be secured using TLS**



107

VI. 하이퍼레저(Hyperledger)

❖ **Ledger/Channel/Smart Contract @ Hyperledger Fabric Architecture**

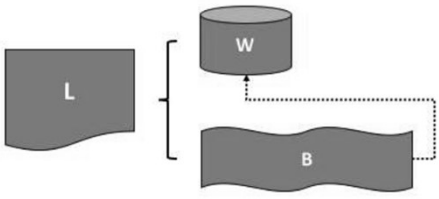


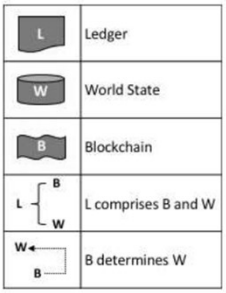
108

VI. 하이퍼레저(Hyperledger)

❖ Ledger

- Ledger L comprises blockchain B and World State W.
- Blockchain B determines World State W.
- Also expressed as: World state W is derived from blockchain B.





james@jslab.kr

https://hyperledger-fabric.readthedocs.io/en/release-1.3/ledger/ledger.html

VI. 하이퍼레저(Hyperledger)

❖ Ledger

- Record of all state transitions of asset
- Immutable, append-only, sequenced, tamper-resistant record.
- Each transaction results in entry to Ledger.
- All participant have replica of ledger
- There is one ledger per channel.
- Each peer maintains a copy of the ledger for each channel of which they are a member.

docker-compose

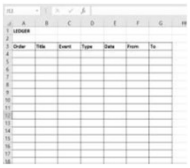
james@jslab.kr

https://hyperledger-fabric.readthedocs.io/en/release-1.3/ledger/ledger.html

VI. 하이퍼레저(Hyperledger)

❖ **Ledger**

- **Chaincode modified Ledger data**
- **Transaction Log**
  - ✓ Records all transactions
  - ✓ Immutable
  - ✓ Can perform only Create and Read
  - ✓ Uses Level DB - Embedded within the peer
- **State database**
  - ✓ Records current state of Asset
  - ✓ Can Perform Create, Read, Update, Delete
  - ✓ New Versioned created on update with Key-Value pair
  - ✓ We can use Couch DB to execute complex queries

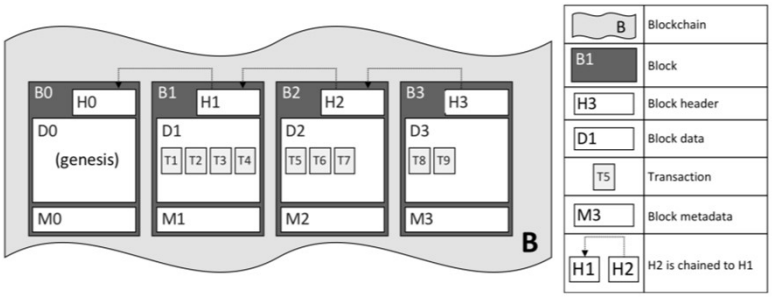


https://hyperledger-fabric.readthedocs.io/en/release-1.3/ledger/ledger.html

VI. 하이퍼레저(Hyperledger)

❖ **Blockchain Structure**

- **In the diagram, we can see that block B2 has a block data D2 which contains all its transactions: T5, T6, T7.**
- **Most importantly, B2 has a block header H2, which contains a cryptographic hash of all the transactions in D2 as**
- **well as with the equivalent hash from the previous block B1. In this way, blocks are inextricably and immutably linked to each other, which the term blockchain so neatly captures!**

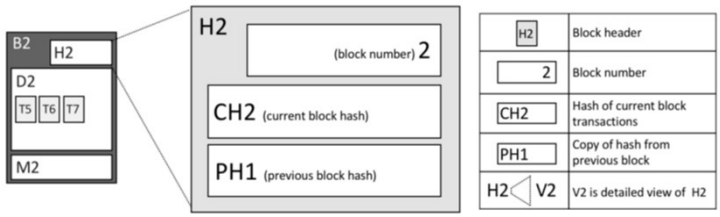


https://hyperledger-fabric.readthedocs.io/en/release-1.3/ledger/ledger.html

VI. 하이퍼레저(Hyperledger)

❖ Blockchain Blocks

- Block Header: This section comprises three fields, written when a block is created.
- Block number: An integer starting at 0 (the genesis block), and increased by 1 for every new block appended to the blockchain.
- Current Block Hash: The hash of all the transactions contained in the current block.
- Previous Block Hash: A copy of the hash from the previous block in the blockchain.

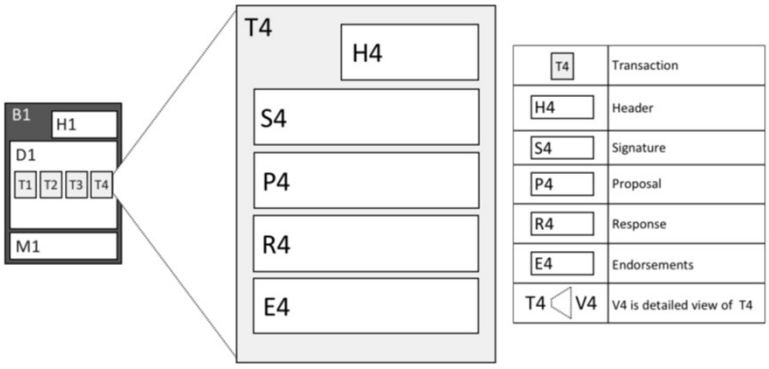


https://hyperledger-fabric.readthedocs.io/en/release-1.3/ledger/ledger.html

113

VI. 하이퍼레저(Hyperledger)

❖ Transactions

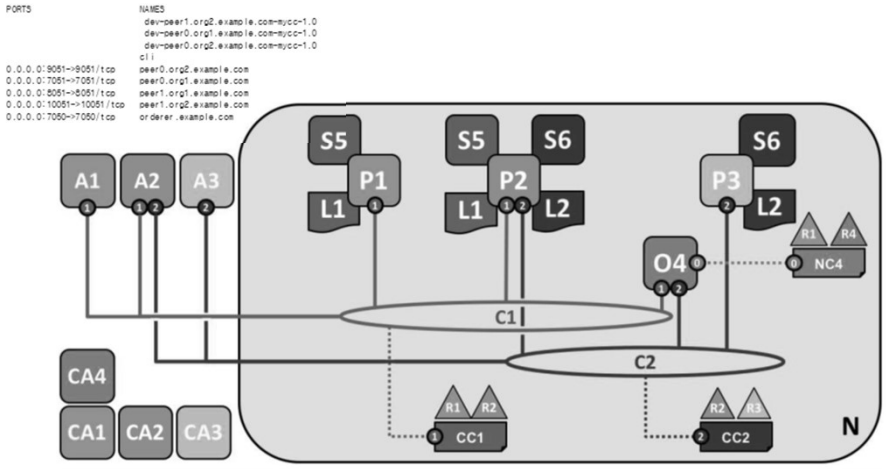


https://hyperledger-fabric.readthedocs.io/en/release-1.3/ledger/ledger.html

114

VI. 하이퍼레저(Hyperledger)

❖ Blockchain Network
❖ The sample network for 4 organizations



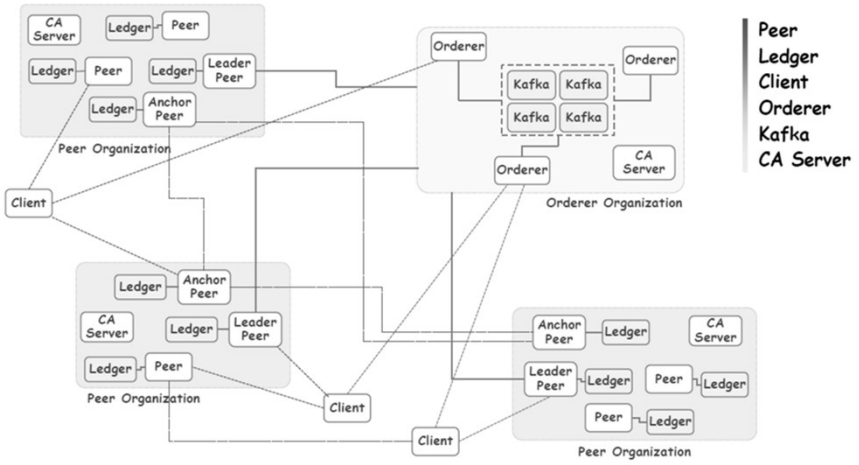
https://hyperledger-fabric.readthedocs.io/en/release-1.4/network/network.html

JS Lab

115

VI. 하이퍼레저(Hyperledger)

❖ Fabric Network / Software Architecture

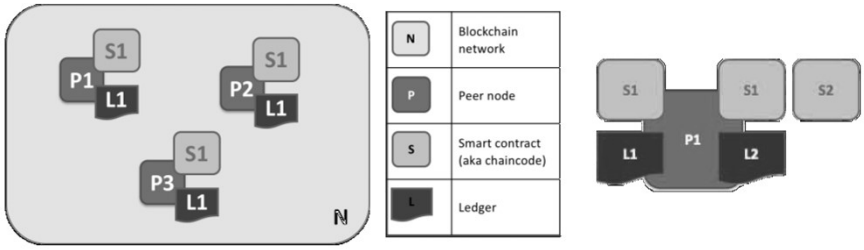


JS Lab

116

VI. 하이퍼레저(Hyperledger)

❖ Peers

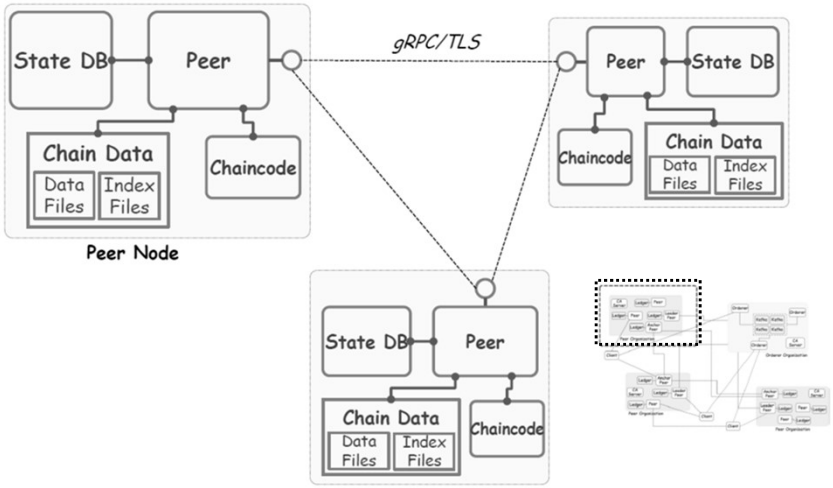


https://hyperledger-fabric.readthedocs.io/en/release-1.3/ledger/ledger.html

117

VI. 하이퍼레저(Hyperledger)

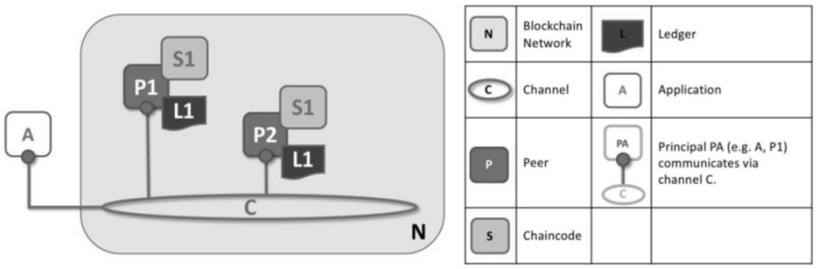
❖ Peer Organization



118

VI. 하이퍼레저(Hyperledger)

❖ Peers and Channels



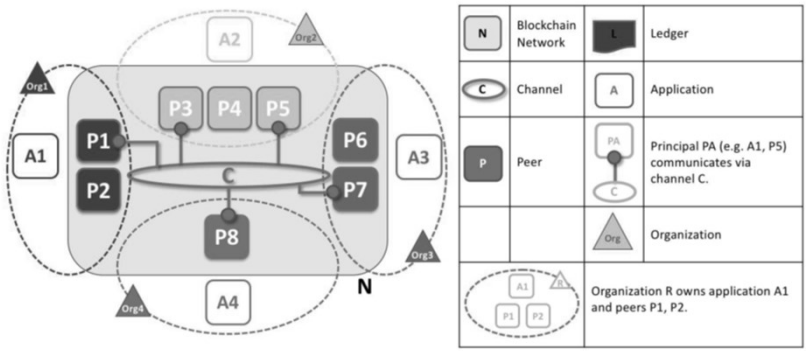
https://hyperledger-fabric.readthedocs.io/en/release-1.3/ledger/ledger.html

james@jslab.kr

119

VI. 하이퍼레저(Hyperledger)

❖ Peers and Organizations

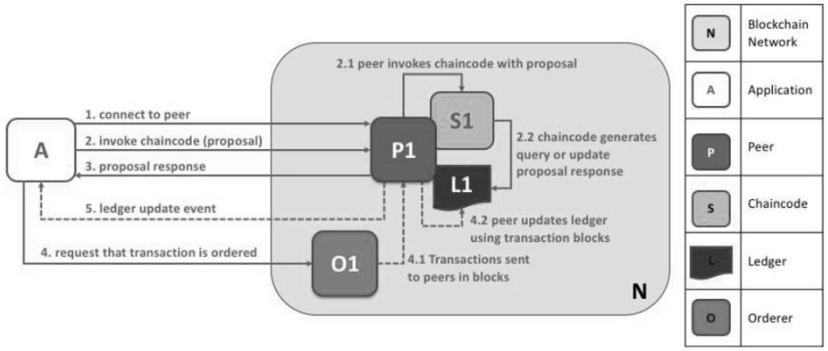


james@jslab.kr

120

VI. 하이퍼레저(Hyperledger)

❖ Transaction Flow

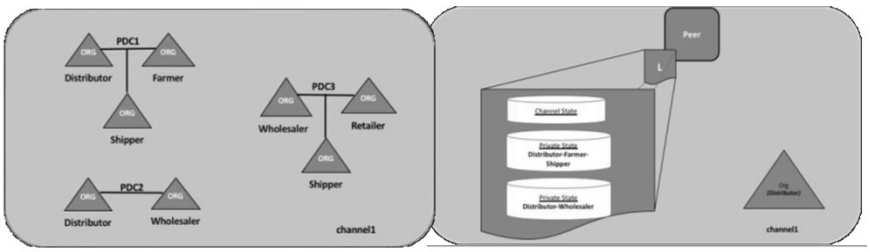


121

VI. 하이퍼레저(Hyperledger)

❖ Private Data – Use Case

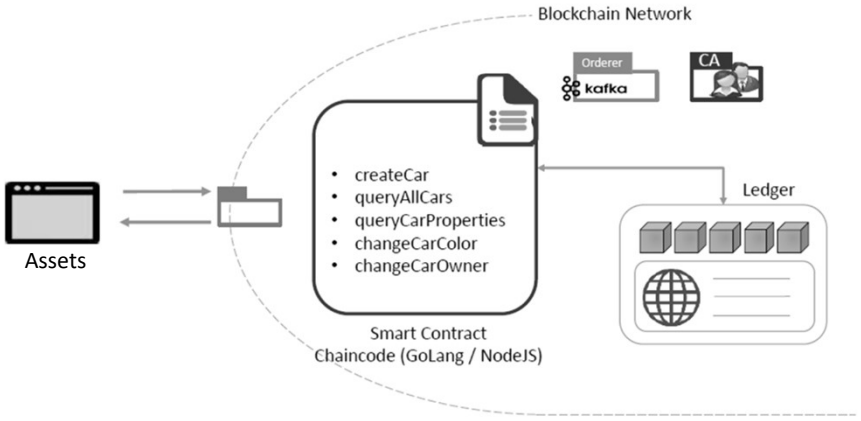
- A Farmer selling his goods abroad
- A Distributor moving goods abroad
- A Shipper moving goods between parties
- A Wholesaler purchasing goods from distributors
- A Retailer purchasing goods from shippers and wholesalers



122

VI. 하이퍼레저(Hyperledger)

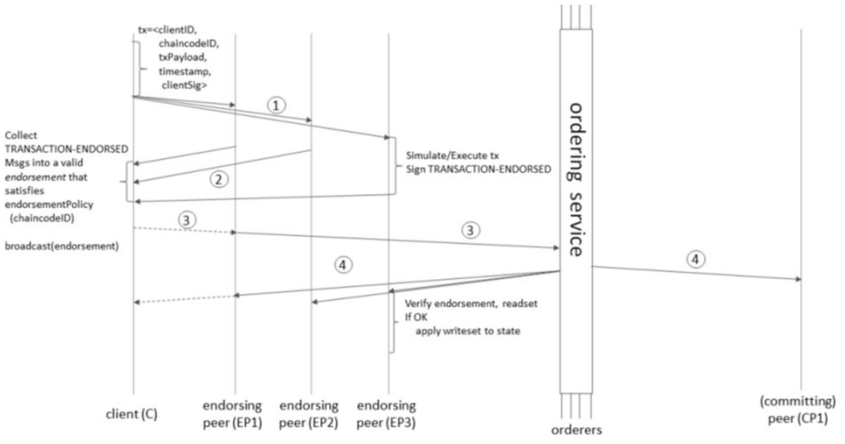
❖ **Fabric Example : Smart Contract Functions**



123

VI. 하이퍼레저(Hyperledger)

❖ **Transaction Proposal to Commit**



124

VI. 하이퍼레저(Hyperledger)

❖ 하이퍼레저 Sawtooth

- PoET 컨센서스 블록체인 - 현재 v1.1 (Bumper)
- Intel의 기부 (Intel Distributed Ledger)
- 기존 산업에 블록체인을 도입해 활용 하도록 하는 프로젝트
- 분산 원장 구축, 배포 및 실행 목적의 모듈식 플랫폼
- 자원 소비를 최소화하고 대규모 분산 유효성 검사를 진행.
- PoET (Proof of Elapsed Time) 기반의 컨센서스
  - ✓ 인텔의 SGX(Secure Guard Extensions) 기술을 기반으로 구현
- Permissioned(허가형)과 Permissionless(참여형) 모두 지원
- 병렬 처리(Parallel Transaction Processing) 지원
- 아파치 2.0 라이선스

james@jslab.kr

https://bitcoinmagazine.com/articles/hyperledger-releases-sawtooth-10/

VI. 하이퍼레저(Hyperledger)

❖ 하이퍼레저 Sawtooth 1.0

- **On-chain governance –** Utilizing smart contracts to vote on blockchain configuration settings such as the allowed participants and smart contracts.
- **Advanced transaction execution engine –** Processing transactions in parallel to accelerate block creation and validation.
- **Support for Ethereum –** Running Solidity smart contracts and integrating with Ethereum tooling.
- **Dynamic consensus –** Upgrading or swapping the blockchain consensus protocol on the fly as networks grow, enabling the integration of more scalable algorithms as they are available.

- 인텔이 만들고 있는 프라이빗 블록체인 플랫폼인 Sawtooth LAKE의 경우 인텔 CPU에서만 생성할 수 있는 데이터를 가지고 합의하는 확률적 알고리즘인 PoET 합의 알고리즘을 사용

james@jslab.kr

https://bitcoinmagazine.com/articles/hyperledger-releases-sawtooth-10/

VI. 하이퍼레저(Hyperledger)

❖ 하이퍼레저 이로하(Iroha)

- 간편한 자산 생성/관리를 위한 블록체인
- 분산 원장 기술을 필요로 하는 인프라 프로젝트를 간단하게 통합할 수 있도록 설계된 비즈니스 블록 체인 체계.
- 간편한 자산 생성/관리를 위한 블록체인
- Soramitsu, Hitachi, NTT 데이타, Colu등 기여
- C++ 사용자를 위한 개발 환경 (C++ 라이브러리)
- 모바일과 웹을 위한 인프라 제공 (iOS, Android, JavaScript 지원)
- YAC(Yet Another Consensus) 컨센서스 알고리즘 : 블록해시에 대한 투표
- 아파치 2.0 라이선스

james@jslab.kr

https://github.com/hyperledger/iroha http://iroha.readthedocs.io/en/latest/getting_started/

127

VI. 하이퍼레저(Hyperledger)

❖ 하이퍼레저 버로우(Burrow)

- 이더리움 Dapp 지원 프레임워크(승인 가능한 스마트 계약 장치)
- 2014년 12월에 처음 출시된 Burrow는 모듈식 블록 체인 고객에게 이더리움 가상 기계(EVM)에 내장된 권한 있는 스마트 계약을 해석.
- 모낙스(Monax)가 제안하여 인텔(Intel)이 공동 지원
- 이더리움 진영과 하이퍼레저 진영에 공존하는 프로젝트
- 3개의 주요 모듈
  - ✓ 컨센서스 엔진 : BFT(Byzantine Fault Tolerant) 알고리즘
  - ✓ 허가형(permissioned) EVM(Ethereum Virtual Machine) : 가스(gas)는 자동 충전됨
  - ✓ RPC 게이트웨이(gateway) : REST와 JSON-RPC 제공아파치 2.0 라이선스
- 아파치 2.0 라이선스

james@jslab.kr

https://github.com/hyperledger/burrow

128

VI. 하이퍼레저(Hyperledger)

❖ 하이퍼레저 인디(Indy)

- 분산 신원 증명 프레임웍
- 자주적인(self-sovereign) 신원 증명 생태계 프레임워크
- 소브린(sovrin) 파운데이션이 주도 (https://sovrin.org/)
  - ✓ 인터넷 환경에서의 신원(identity)를 제공하는 플랫폼
  - ✓ IBM, ATB 파이낸셜, T-랩스, 에버님(Evernym) 등 30여 개 조직 참여
  - ✓ 변조가 어려운 블록체인 네트워크로 인터넷에 신원 계층 추가
  - ✓ 현재 베타 상태로, IBM의 직원 ID 인증을 시범 운영 중
- 상호 운용성을 위해 블록 체인 또는 다른 분산 원장 기반의 독립적인 디지털 개체를 생성하고 활용하기 위한 전용 도구, 라이브러리 및 구성 요소를 제공
- 아파치 2.0 라이선스

james@jslab.kr

https://github.com/hyperledger/indy-node

VI. 하이퍼레저(Hyperledger)

❖ 블록체인 익스플로러(Blockchain Explorer)

- 운영,개발을 위한 툴(Tools)
- 블록체인 런타임에 대한 다양한 정보를 보여주는 웹기반 애플리케이션을 만드는 프로젝트, 네트워크를 모니터링하기 위한 시스템
- 블록 거래, 관련 데이터, 네트워크 정보, 체인 코드, 거래 모음 및 원장에 저장된 기타 관련 정보를 열람, 호출, 배포 또는 쿼리 가능
- IBM과 DTCC이 주도
- 하이퍼레저 Fabric 1.4 지원
- 아파치 2.0 라이선스

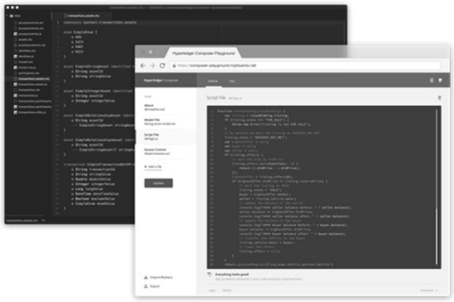
james@jslab.kr

https://github.com/hyperledger/blockchain-explorer

VI. 하이퍼레저(Hyperledger)

❖ 하이퍼레저 컴포저(Composer)

- 운영,개발을 위한 툴(Tools)
- 블록 체인 비즈니스 네트워크 구축 목적의 공동 작업 도구로 스마트 계약 개발 및 분산 원장내의 배포를 가속화 블록체인 애플리케이션 개발을 위한 협업 프레임워크
- Javascript 기반
- 아파치 2.0 라이선스

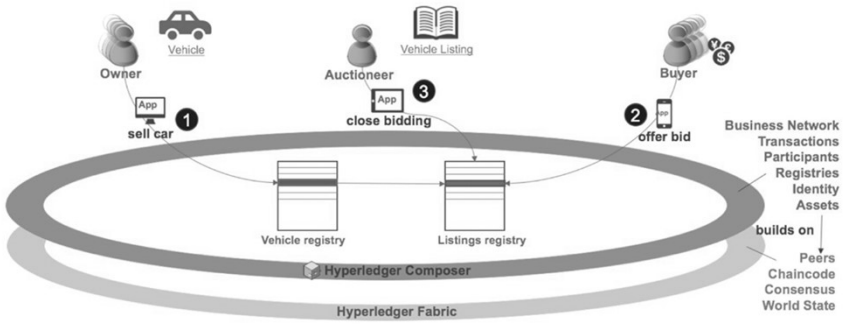


https://github.com/hyperledger/composer

VI. 하이퍼레저(Hyperledger)

❖ 하이퍼레저 컴포저(Composer)

- Building a Blockchain PoC in Ten Minutes Using Hyperledger Composer
- Designing a Blockchain Car Auction



https://www.infoq.com/articles/blockchain-poc-hyperledger/

VI. 하이퍼레저(Hyperledger)

❖ 하이퍼레저 캘리퍼(Caliper)

- 운영, 개발을 위한 툴(Tools), 성능 측정 툴
- 블록체인 성능(Performance) 테스트 툴
- 2018년 5월 현재, 아래의 3가지 블록체인 솔루션 지원
  - ✓ 하이퍼레저 fabric 1.0+
  - ✓ 하이퍼레저 sawtooth 1.0+
  - ✓ 하이퍼레저 Iroha
- 하이퍼레저 컴포우저(Composer)에 대한 성능 측정 지원
- 성공률, 처리량, 트랜잭션 지연(min, max,..), 자원사용(CPU, Memory..)
- 아파치 2.0 라이선스

https://github.com/hyperledger/caliper/blob/master/docs/Composer.md

133

VI. 하이퍼레저(Hyperledger)

❖ 하이퍼레저 캘리퍼(Caliper)

Test	Name	Succ	Fail	Send Rate	Max Latency	Min Latency	Avg Latency	75%ile Latency	Throughput
1	physical-access-network	500	0	10 tps	20.67 s	1.32 s	13.72 s	18.18 s	7 tps

TYPE	NAME	Memory(max)	Memory(avg)	CPU(max)	CPU(avg)	Traffic In	Traffic Out
Process	node bench-client.js(avg)	-	-	NaN%	NaN%	-	-
Docker	dev-peer0.org1.example.co...0.1.0	125.5MB	121.7MB	101.01%	20.61%	4.5MB	3.7MB
Docker	dev-peer0.org2.example.co...0.1.0	121.4MB	115.1MB	102.06%	21.12%	4.5MB	3.6MB
Docker	peer0.org1.example.com	368.4MB	343.5MB	17.73%	11.75%	16.6MB	32.5MB
Docker	peer0.org2.example.com	359.6MB	333.3MB	17.87%	11.66%	16.5MB	32.7MB
Docker	couchdb.org1.example.com	114.5MB	109.2MB	54.88%	32.39%	4.5MB	8.1MB
Docker	couchdb.org2.example.com	117.2MB	110.9MB	53.61%	32.86%	4.5MB	8.1MB
Docker	orderer.example.com	17.7MB	14.7MB	3.64%	1.85%	3.9MB	7.8MB
Docker	ca.org1.example.com	5.3MB	5.3MB	4.37%	0.20%	4.6KB	3.6KB
Docker	ca.org2.example.com	7.3MB	7.3MB	0.00%	0.00%	1.9KB	0B

Physical Access Control Management System Based on Permissioned Blockchain
Department of Computer Science University of Saskatchewan Saskatoon, Canada

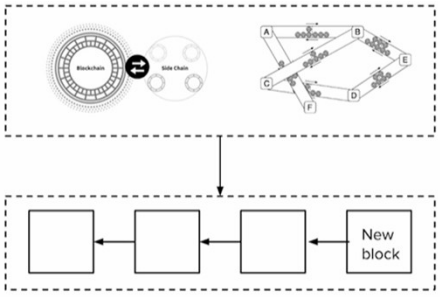
134

VI. 하이퍼레저(Hyperledger)

❖ 하이퍼레저 캘리퍼(Caliper)

- **Off-chain scaling**
- **On-chain scailng**

- Layer 2 scaling refers to pushing computation off the blockchain
  - Off-chain scaling
- Layer 1 scaling refers to changing the blockchain itself
  - On-chain scaling



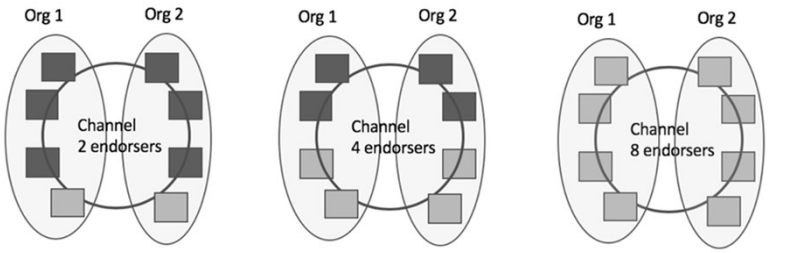
Sources:
https://raiden.network/101.html
https://en.bitcoinwiki.org/wiki/Sidechain

135

VI. 하이퍼레저(Hyperledger)

❖ 하이퍼레저 캘리퍼(Caliper)

- **Hyperledger Fabric 1.3.0**
- **Single Kubernetes cluster**
- **IBM Container Service with worker nodes configured as 4vCPU and 16Gb memory with SSDs**



# endorsers	2	4	8
TPS | 95% (ms)	785.58 | 715	948.2 | 667	1265.5 | 686

https://www.ibm.com/blogs/blockchain/2019/01/answering-your-questions-on-hyperledger-fabric-performance-and-scale/

136

VI. 하이퍼레저(Hyperledger)

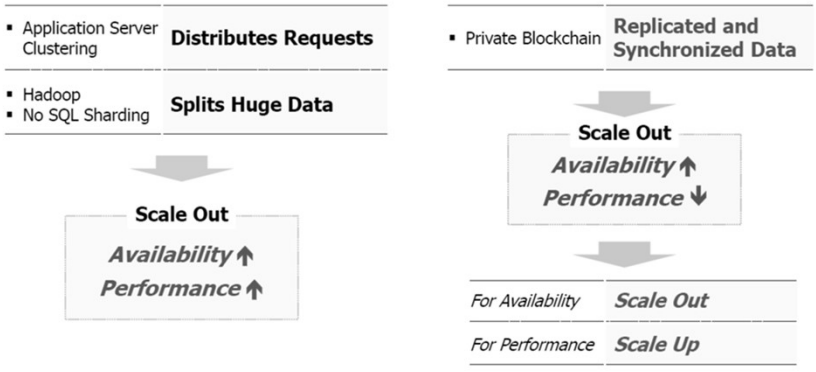
❖ 하이퍼레저 캘리퍼(Caliper)
❖ 성능 관련 요소

- Number of endorsing peers
- Number of channels
- Number of endorsements (endorsement policy)
- Ordering service configuration (block size and frequency)
- Number of organizations
- Ledger database used
- Complexity of chaincode/smart contract execution
- Size of transactions
- Use of mutual TLS for all network traffic
- Number of vCPUs
- Memory allocation
- Disk type and speed
- Network speed
- Multiple datacenter deployment
- CPU speed
- Crypto acceleration

137

VI. 하이퍼레저(Hyperledger)

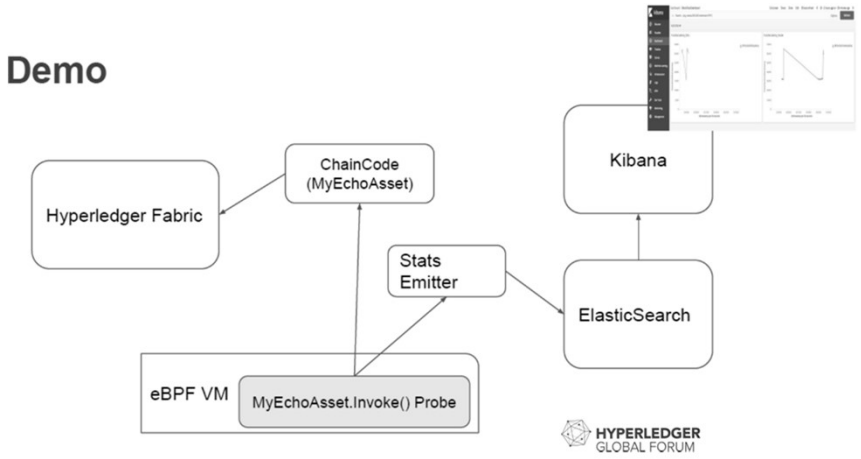
❖ 하이퍼레저 캘리퍼(Caliper)
❖ Scale Out / Scale Up



138

VI. 하이퍼레저(Hyperledger)

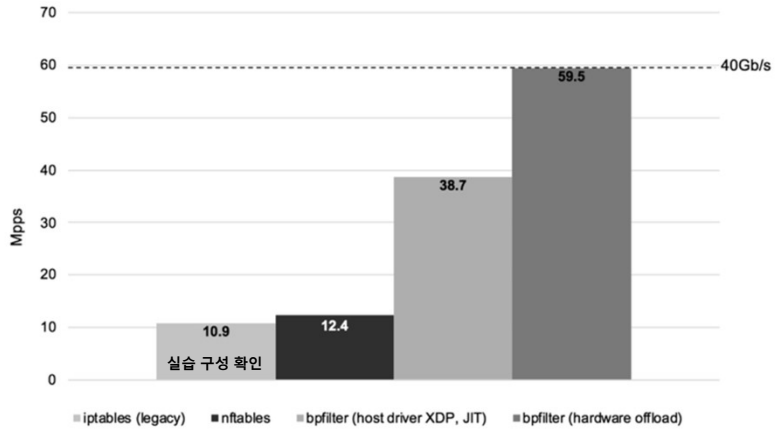
❖ 하이퍼레저 캘리퍼(Caliper)
❖ eBPF (Extended Berkeley Packet Filter) for higher throughput



139

VI. 하이퍼레저(Hyperledger)

❖ eBPF XDP Linux 4.8+
❖ 소프트웨어 기반 성능 가속
❖ 성능 가속 전용 하드웨어 발전 (스마트 NIC, HCI, 컨테이너 가속 Storage 등등)



https://www.netronome.com/blog/frnog-30-faster-networking-la-francaise/

140

VI. 하이퍼레저(Hyperledger)

❖ 하이퍼레저 캘리퍼(Caliper)
❖ eBPF! (Extended Berkeley Packet Filter)

- Highly programmable
- Highly granular
- Can be inline, but doesn't require code modification
- Ideal for blockchain and smart contracts
  - ✓ Linux Perf can run eBPF
  - ✓ iovisor bcc is preferred over Linux Perf
- Brings in visibility at userland function level
- Couple this with call graphing
- Enables measurement of -
  - ✓ Latency of functions in a smart contract
  - ✓ Resource utilization of functions in a smart contract

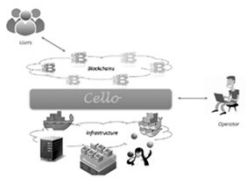
james@jslab.kr

141

VI. 하이퍼레저(Hyperledger)

❖ 하이퍼레저 첼로(Cello)

- 블록체인 운영 관리 툴
- 맞춤형 '서비스' 배포 모델을 블록 체인 생태계에 적용하여 블록 체인 생성, 관리 및 종료 시간을 최소화하기 위한 툴
- 베어메탈, VM, 컨테이너 상의 블록체인 관리를 위한 툴
- IBM, 소라미추(soramitsu), 화웨이(Huawei), 인텔(Intel)이 후원
- 대시보드를 통한 시스템 상태를 확인하고, 자원 확장 등 지원
- 생성/시작/중단/삭제/유지 등 블록체인 라이프사이클관리
- 현재 하이퍼레저 Fabric 지원. 향후 타 블록체인 지원 예정
- 이기종 환경(x86, Power, ..) 지원
- 아파치 2.0 라이선스



https://github.com/hyperledger/cello

james@jslab.kr

142

VI. 하이퍼레저(Hyperledger)

❖ 하이퍼레저 퀄트(Quilt)

- Ledger 간 연동 프로토콜
- 하이퍼레저 Quilt는 주로 지급 프로토콜인 분산 원장과 비분산 원장 사이의 가치 이전 목적으로 설계된 ILP를 구현하여 원장 시스템의 상호 운용성을 극대화 시키는 툴
- ILP(Inter-Ledger Protocol) 구현으로 블록체인간 상호 운용성 제공
- NTT 데이터(Data)와 리플(Ripple) 제안
- 기본적으로 지불(payment) 기능 제공 예정
- 분산원장과 기존 원장을 포함한 네트워크 상의 가치(value) 이전 지원
- OER(Octet Encoding Rules) 기반의 암호화/복호화 지원
- 리눅스 파운데이션 내의 2개의 ILP 구현체 중 하나
- Quilt – Java 기반, interledger.js–Javascript기반
- 아파치 2.0 라이선스

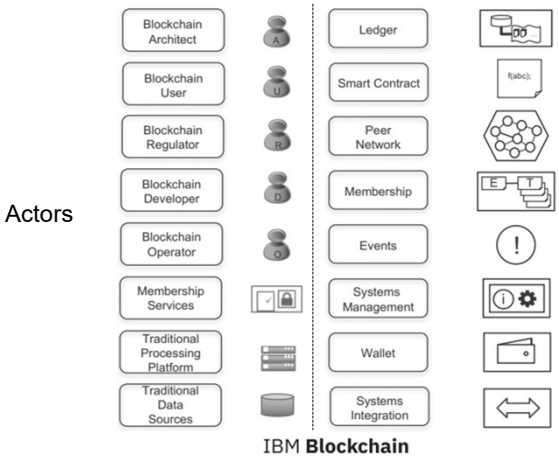
https://github.com/hyperledger/quilt

143

VI. 하이퍼레저(Hyperledger)

❖ 인프라 운영을 위한 솔루션 구성 요소 (IBM)

- Actors
- Components



144

VI. 하이퍼레저(Hyperledger)

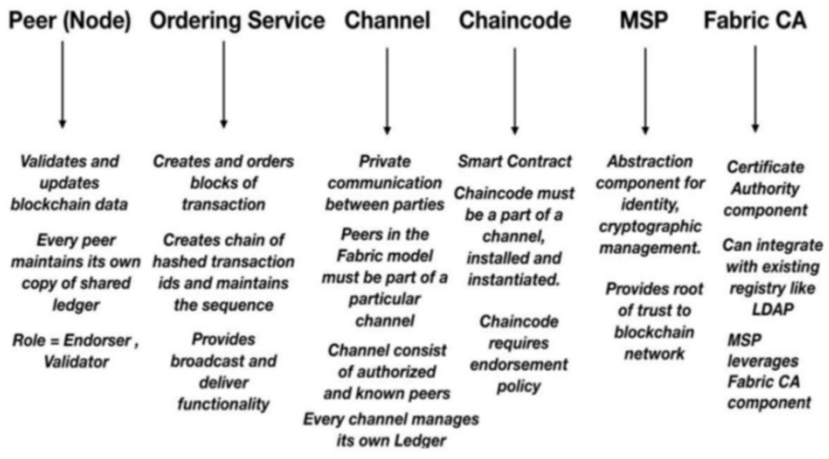
❖ 요약

- 비교 (Hyperledger : Bitcoin : Ethereum)

	Bitcoin	Ethereum	Hyperledger Frameworks
암호 화폐 기반	Yes	Yes	No
허가 (Permissioned)	No	No	Yes (in general)*
익명 사용	Yes	No	No
감사 가능	Yes	Yes	Yes
불변의 원장	Yes	Yes	Yes
모듈화	No	No	Yes
스마트 계약	No	Yes	Yes
컨센서스 프로토콜	PoW	PoW	Various**

VI. 하이퍼레저(Hyperledger)

❖ 기능 요약 (참고)



source: Building Blockchain Apps using Ethereum & Hyperledger@Udemy

james@jslab.kr

I. **개요**
II. **퍼블릭 블록체인** (Public Blockchain)
III. **스마트 계약** (Smart Contract)
IV. **프라이빗 블록체인** (Private Blockchain)
V. **컨소시엄 블록체인** (Consortium Blockchain)
VI. **하이퍼레저** (Hyperledger)
VII. **컨센서스** (Consensus)
VIII. **사례** (Use Cases)
❖ **실습** (별도 교재)

147

james@jslab.kr

VII. 컨센서스 (Consensus)

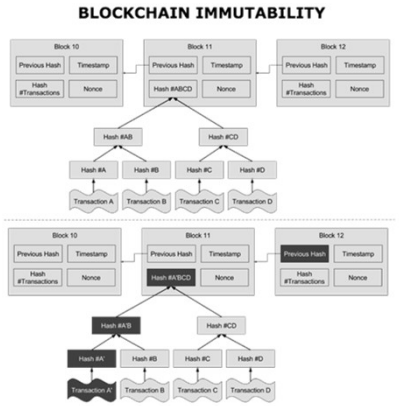
http://www.plays-in-business.com/consensus-decisioning-how-to-find-minimal-viable-decisions/

148

VII. 컨센서스 (Consensus)

❖ 컨센서스 알고리즘 (Consensus Protocol/ Consensus Mechanism)

- 클러스터링된 인프라의 오류 발생 환경에서 서비스 지속 가능한 알고리즘
- 정상 작동 노드가 과반수 이상이면 분산원장의 변경을 서비스를 지속 가능



Merkle Tree

- 바이너리 해시 트리
- 각 데이터의 해시를 저장하는데 사용하는 데이터 구조

거래별 관련 해시

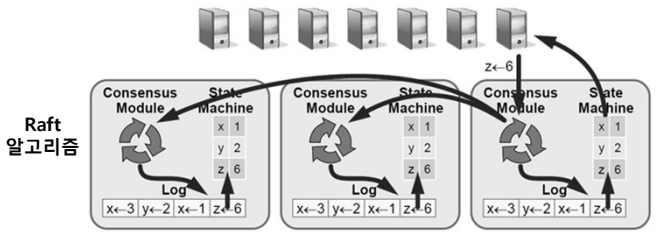
- 전(前) 트랜잭션 해시
- 동일 블록 내의 거래 모음 해시
- 자신의 거래 해시

james@jslab.kr

VII. 컨센서스 (Consensus)

❖ Consensus Algorithms

- 분산 시스템에 참여하고 있는 모든 노드가 같은 결과 값을 결정해야 함
- 합의 문제는 분산 시스템의 신뢰도를 보장하기 위해 나온 개념으로 블록체인이 나오기 전부터 존재
- 결정된 데이터는 특정 Process에 의해 제안된 것이어야 함
- 모든 시스템의 상태는 0이나 1로 결정 (모두 1인지 0인지 판단 할 수 있어야 함)
- 악의적인 노드가 존재하더라도 신뢰도 있는 시스템 제공 보장



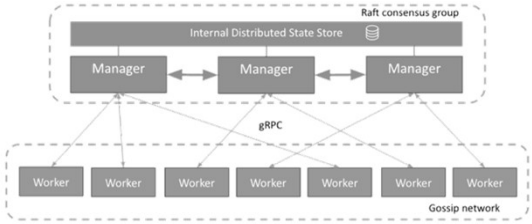
- Leader Election
- Log Replication
- Safety

james@jslab.kr

VII. 컨센서스 (Consensus)

- ❖ 스웜모드 아키텍처 토폴로지
- ❖ 스웜(Swarm) Manager 는 3개 or 5개 or 7개등 홀수 권장

스웜 매니저간들의 통신은 항상 쿼롬(Quorum) 가능



- /var/lib/docker/swarm/docker.state.json

{"LocalAddr":"","RemoteAddr":"192.168.99.118:2377","ListenAddr":"0.0.0.0:2377","AdvertiseAddr":""}

- /var/lib/docker/swarm/state.json

[{"node_id":"9c5eqant0s2w7arlfk47tkxm0","addr":"192.168.99.118:2377"},
{"node_id":"bexym9a2cxbd60ow40xibycw5","addr":"192.168.99.115:2377"},
{"node_id":"cviejn6myjln6s4ysw4wg59rn","addr":"192.168.99.119:2377"}]

'docker info'의 swarm 정보 표시(예)

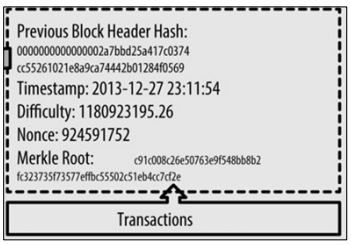
```
Swarm: active
 NodeID: kj4l7bnzmr8yphsmln33zs7a8
 Is Manager: true
 ClusterID: tzzk07zwhg8axv0jayf2u58yd
 Managers: 3
 Nodes: 8
 Orchestration:
  Task History Retention Limit: 5
 Raft:
  Snapshot Interval: 10000
  Number of Old Snapshots to Retain: 0
  Heartbeat Tick: 1
  Election Tick: 3
 Dispatcher:
  Heartbeat Period: 5 seconds
 CA Configuration:
  Expiry Duration: 3 months
  External CAs:
    cfssl: https://192.168.99.131:12381/api/v1/cfssl/sign
    cfssl: https://192.168.99.129:12381/api/v1/cfssl/sign
    cfssl: https://192.168.99.130:12381/api/v1/cfssl/sign
 Node Address: 192.168.99.129
 Manager Addresses:
  192.168.99.129:2377
  192.168.99.130:2377
  192.168.99.131:2377
```


VII. 컨센서스 (Consensus)

❖ Proof of Work (PoW)

- "작업 증명"(Proof of Work, POW)이라는 용어는 1999년 Markus Jakobsson와 Ari Juels의 논문에서 의해 처음 만들어져 공식화
- 작업을 요구함으로써 서비스 거부(DoS) 공격과 기타 서비스 악용(예: 네트워크 상의 스팸)을 단념하게 만들기 위한 경제적인 수단
- 마이닝(mining) 프로세스로 알려져 있으며 증명(Proof)은 어렵게 생성 되지만 검증(Verify) 은 쉽게 함
- 마이닝 성공을 위한 인센티브를제공
- PoW consensus algorithm의 에너지 과다 사용에 대한 비판 시각이 있음.

비트코인 블록 구조
(출처 : mastering bitcoin)



5,000,000 TH/s(1 TH/s = 초당1,000,000,000,000번의 해시연산) 이상의 해시 파워가 필요

VII. 컨센서스 (Consensus)

❖ Proof of Stake (PoS)

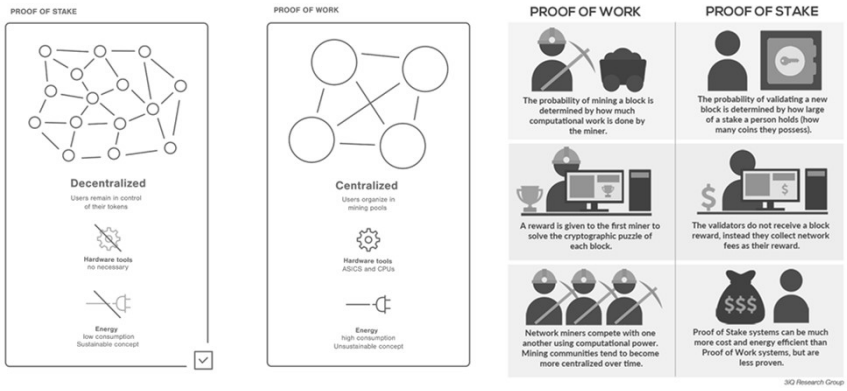
- 컴퓨팅 파워 낭비가 아닌 자신이 가진 지분(stake)을 통해 블록을 생성
- 자신이 가지고 있는 지분(Stake)과 지분이 생성된 날짜에 의해 결정되며, 한번 블록 생성을 위해 사용된 지분의 날짜는 초기화
- 합의 노드가 변조등 악의적인 행동을 했을 경우 프로토콜에 의해 몰수당하거나 '슬래싱(slashing)' 될 수 있음
- PoW에서는 합의 노드를 채굴자(miner)라고 하지만 PoS에서는 합의 노드를 검증인(validator)이라고 함
- 검증인의 투표권(voting power) 내지 가중치(weight)는 보유 지분 또는 위임된 지분의 총량과 비례
- PoS algorithm은 컴퓨팅 자원을 절약

james@jslab.kr

153

VII. 컨센서스 (Consensus)

❖ Proof of Work(PoW)와 Proof of Stake (PoS)



https://hackernoon.com/consensus-mechanisms-explained-pow-vs-pos-89951c66ae10

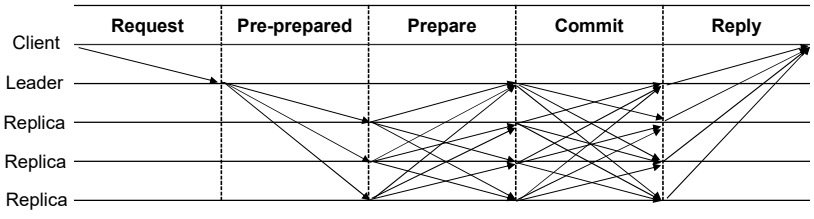
james@jslab.kr

154

VII. 컨센서스 (Consensus)

❖ **Practical Byzantine Fault-tolerance (PBFT)**

- 분산시스템이 약속된 행동을 하지 않는 비잔틴 노드가 존재할 수 있는 비동기 시스템일 때 해당 분산시스템에 참여한 모든 노드가 성공적으로 합의를 이룰 수 있도록 개발된 합의 알고리즘
  1) 리더가 클라이언트들의 요청을 수집/정렬하고 실행 결과와 함께 다른 노드들에 전파
  2) 메시지를 받은 노드들은 다른 노드들에서 받은 메시지를 다시 나머지 노드들에게 전파
  3) 모든 노드는 자신이 다른 노드에서 가장 많이 받은 같은 메시지(정족수 이상의)가 무엇인지 다른 노드들에게 전파
  4) 1) 2) 3)의 과정이 끝나면 모든 노드들은 정족수 이상이 동의한, 즉 합의를 이룬 같은 데이터를 가짐



https://theintelligenceofinformation.wordpress.com/2017/02/15/practical-byzantine-fault-tolerance-consensus-talk-by-miguel-castro/

155

VII. 컨센서스 (Consensus)

❖ **비교**

- **Consensus** 는 여러가지가 있으며 네트워크의 요구에 따라 목표 기능을 구현

Types of Blockchain	Example of Blockchain	Consensus Algorithm	Nature
Public Blockchains	Bitcoin, Ethereum, Litcoin etc	PoW, PoS, DPoS	Open and decentralized
Federated Blockchains	R3, B3I, EWF	No	Controlled and Restricted
Private Blockchains	Company Internal	PBFT, RAFT	Restricted
Permisioned Blockchain	Hyperledger, Ripple	PBFT	Closed and restrcited

Characteristics	PoW	PoS	DPos	PBFT	RAFT
Byzantine Fault Tolerance	50%	50%	50%	33%	N/A
Crash Fault Tolerance	50%	50%	50%	33%	50%
Verification Speed	>100s	<100s	<100s	<10s	<10s
Throughput(TPS)	<100	<1000	<1000	<2000	>10k
Scalability	strong	strong	strong	weak	weak

https://medium.com/coinmonks/know-which-blockchain-or-dlt-platform-works-well-within-your-usecase-comparison-of-different-a8dc34782af3

156

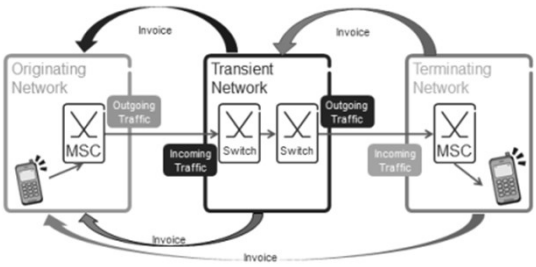
I. 개요
II. 퍼블릭 블록체인 (Public Blockchain)
III. 스마트 계약 (Smart Contract)
IV. 프라이빗 블록체인 (Private Blockchain)
V. 컨소시엄 블록체인 (Consortium Blockchain)
VI. 하이퍼레저 (Hyperledger)
VII. 컨센서스 (Consensus)
VIII. 사례 (Use Cases)
❖ 실습 (별도 교재)

james@jslab.kr

VIII. 사례 (Use Cases)

❖ **Inter-carrier Charges @ Telecom**

- **Network operators collect and store detailed information in a record known as Call Detail Record (CDR) about every call ever attempted, whether completed or not. A typical CDR captures data such as calling and called party phone numbers, duration of the call, CDR unique ID, a timestamp for each activity, the ID of the equipment that handled the call, the result of the call, and so on. Interconnect partners share CDRs for the purpose of "verifying" cross charges and settling balances. This verification process is cumbersome, inefficient, lengthy, costly, and error-prone. Missing CDRs and discrepancies in CDRs are very common problems.**

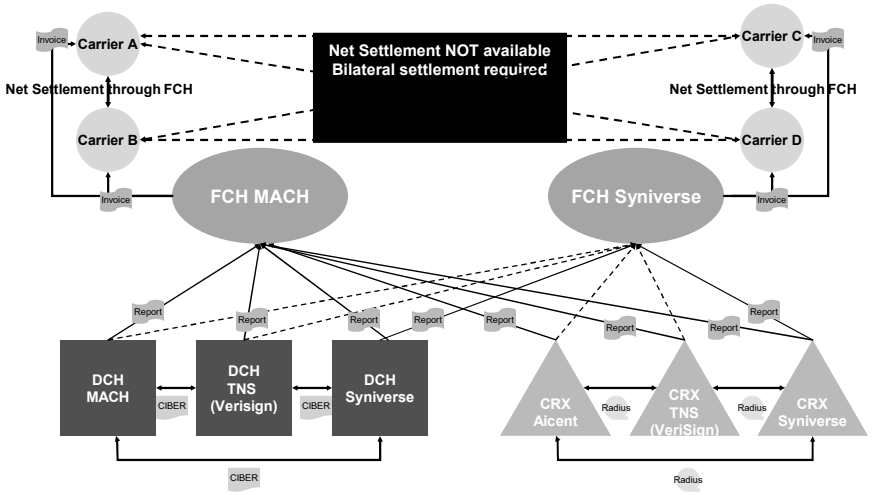


https://telecoedge.com/2010/01/28/interconnect-billing/

james@jslab.kr

VIII. 사례 (Use Cases)

❖ FCH/DCH for Inter-carrier Charges @ Telecom



159

VIII. 사례 (Use Cases)

❖ Why Blockchain for Inter-carrier Charges @ Telecom

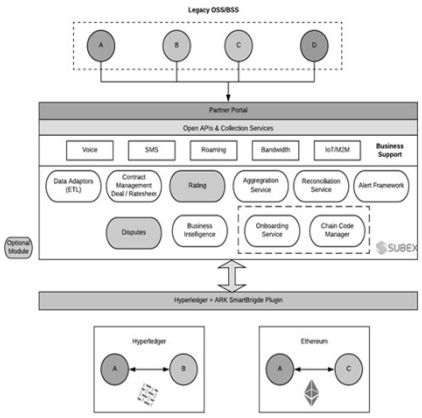
- Network operators would store CDR data related to the Interconnect cross-charging and billing into the Blockchain in real time.
- It eliminates the need for data clearing houses and makes the reconciliation and settlement process simple and error-free. It also helps fighting and preventing fraud.
- Certified charging data can also be stored on the Blockchain, which would eliminate the need for financial clearing houses.
- Network operators dedicate resources just to negotiate Interonnect charges and clearing agreements with other network operators.

160

VIII. 사례 (Use Cases)

❖ **Business Architecture for Inter-carrier Charges @ Telecom**

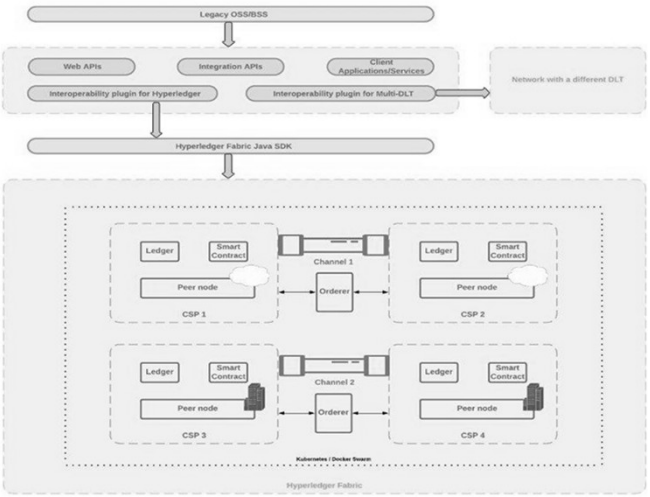
- **Proposed solution is a event agnostic platform that can manage Voice, SMS, Roaming, IoT, Content or any other event settlement scenarios making it is true convergent solution.**



161

VIII. 사례 (Use Cases)

❖ **Business Architecture for Inter-carrier Charges @ Telecom**



162

VIII. 사례 (Use Cases)

❖ **Business Architecture for Inter-carrier Charges @ Telecom**

- **The proposed solution will complement existing legacy BSS/OSS solutions deployed by CSPs by enabling blockchain based settlement capabilities to drastically reduce complexities of dispute management and settlement processes.**
- **A distributed ledger (also called a shared ledger or distributed ledger technology or DLT)**

james@jslab.kr

163

VIII. 사례 (Use Cases)

❖ **Partner Portal for Inter-carrier Charges @ Telecom**

- **Self-Onboarding of new Partners**
- **Ratesheets Smart Contract creation interface**
- **Pricelist Ingestion**
- **Payment & Collections**
- **Bilateral deal Smart Contract creation interface**
- **Publish Reports and Dashboard of reconciliation including discrepancies**
- **Disputes Management**

james@jslab.kr

164

VIII. 사례 (Use Cases)

❖ **Partner Portal for Inter-carrier Charges @ Telecom**

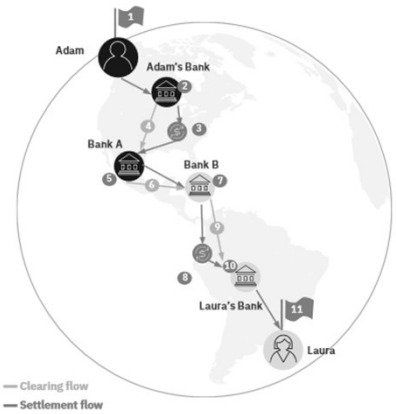
- **Self-Onboarding of new Partners**
- **Ratesheets Smart Contract creation interface**
- **Pricelist Ingestion**
- **Payment & Collections**
- **Bilateral deal Smart Contract creation interface**
- **Publish Reports and Dashboard of reconciliation including discrepancies**
- **Disputes Management**

165

VIII. 사례 (Use Cases)

❖ **International Payment System**

Current international payment system today

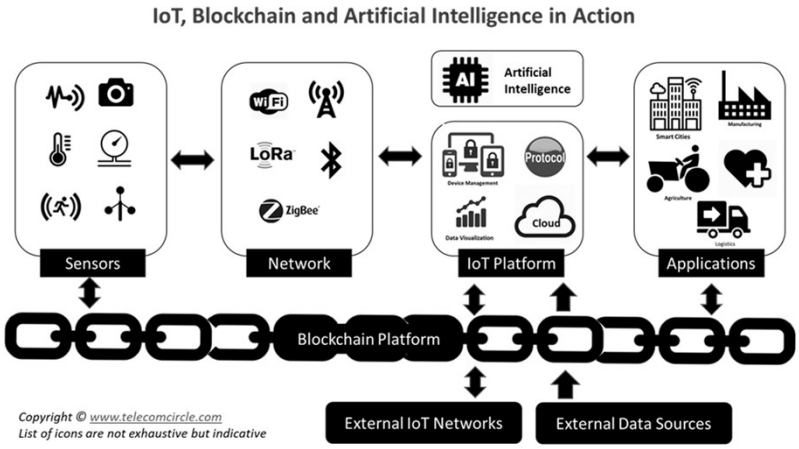


https://www-01.ibm.com/support/docview.wss?uid=ibm10737915

166

VIII. 사례 (Use Cases)

❖ **IoT, Blockchain, AI**

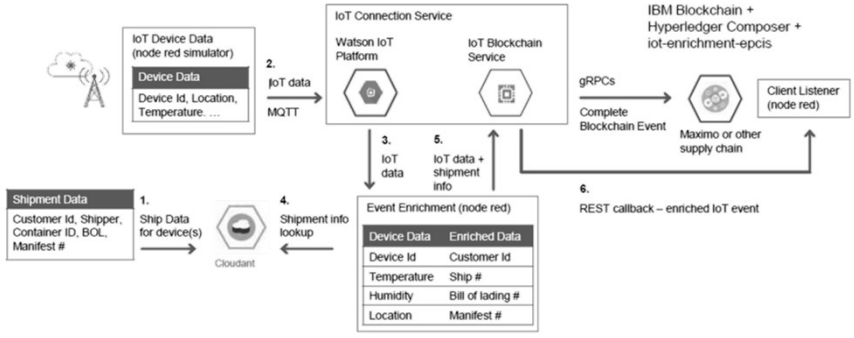


167

VIII. 사례 (Use Cases)

❖ **Objective: Provide an IoT framework that enables IoT providersto easily integrate business data with IoT events for complete business transactions. (IBM Watson)**

IoT blockchain enabled Supply Chain

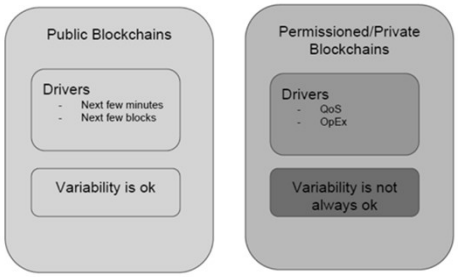


168

VIII. 사례 (Use Cases)

❖ **Billing**

- **Enterprise billing must be multi variate. Examples:**
- **Specific function chain latency**
- **Partnership service charges**
- **IP valued smart contract services**
- **Demand based contract/service pricing**
- **Resource utilization based pricing**
- **Sub-licensing or multi party service provider based pricing**
- **Blockchain frameworks in general can help build Automated Billing Policy Engines**



169

I. 개요
II. 퍼블릭 블록체인 (Public Blockchain)
III. 스마트 계약 (Smart Contract)
IV. 프라이빗 블록체인 (Private Blockchain)
V. 컨소시엄 블록체인 (Consortium Blockchain)
VI. 하이퍼레저 (Hyperledger)
VII. 컨센서스 (Consensus)
VIII. 사례 (Use Cases)
❖ 실습 (별도 교재)

170

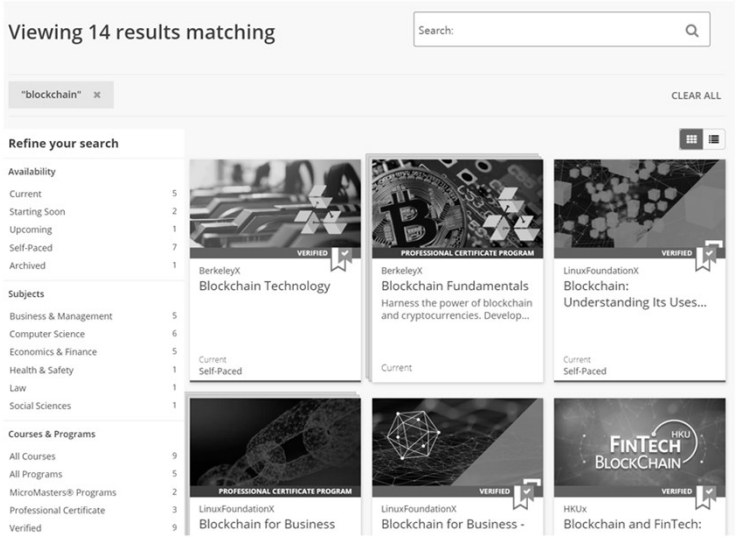
james@jslab.kr

171

james@jslab.kr

Backup

❖ **Blockchain @ edx: 14 classes**

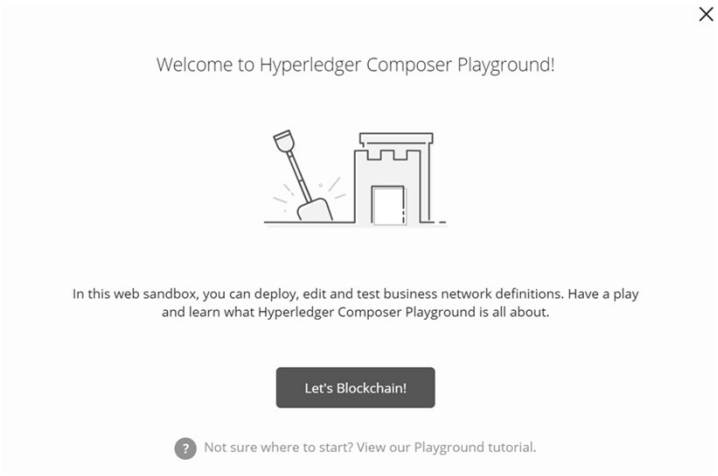


https://www.edx.org/course?search_query=blockchain

172

Backup

https://composer-playground.mybluemix.net/login



173

Fabric Example : Pre requisites – Fabric Network

55

3

$ git clone -b master https://github.com/hyperledger/fabric-samples.git
$ cd fabric-samples
$ git checkout {TAG}

Please execute the following command from within the directory into which you will extract the platform-specific binaries:

$ curl -sSL https://goo.gl/6wtTN5 | bash -s 1.1.0

If you get an error running the above curl command, you may have too old a version of curl that does not handle redirects or an unsupported environment.

$ export PATH=<path to download location>/bin:$PATH

https://hyperledger-fabric.readthedocs.io/en/latest/samples.html

174

Fabric Example : Install the Fabcar Example

55

3

1. Set up the Development Environment
2. Learn the parameters of Smart Contract
3. Develop Apps that can Query and Update the Ledger

https://hyperledger-fabric.readthedocs.io/en/latest/write_first_app.html

175

Fabric Example : Install the Fabcar Example

55

3

$ cd fabric-samples/fabcar && ls *.js

You will see for Node JS Programs as follows

enrollAdmin. js Invoke.js query.js registerUser.js

Do a cleanup before u start installing the Sample App in the Ledger

$ docker rm -f $(docker ps -aq)
$ docker network prune
$ docker rmi dev-peer0.org1.example.com-fabcar-1.0-5c906e402ed29f20260ae42283216aa75549c571e2e380f3615826365d8269ba

$ npm install

$ ./startFabric.sh

https://hyperledger-fabric.readthedocs.io/en/latest/write_first_app.html

176