



블록체인DApp설계

3. 이더리움 클라이언트

경기대학교 AI컴퓨터공학부 이재흥
jhlee@kyonggi.ac.kr

CONTENTS

PRESENTATION



- 이더리움 클라이언트
- 이더리움 네트워크
- 이더리움 클라이언트 실행
- 이더리움 기반 블록체인의 첫 번째 동기화
- 원격 이더리움 클라이언트



이더리움 클라이언트

- 이더리움 클라이언트
 - 이더리움 사양을 구현하고 다른 이더리움 클라이언트와 피어투피어 네트워크를 통해 통신하는 소프트웨어 애플리케이션
 - 기준 사양과 표준 통신 프로토콜을 준수한다면, 서로 다른 이더리움 클라이언트들끼리 상호운용(interoperate)이 가능
 - 황서(yellow paper)라는 공식 사양에 의해 정의됨
 - 비트코인의 경우 비트코인 코어(Core) 구현이 기준



이더리움 네트워크

- 이더리움 프로토콜의 여섯 가지 기본 구현
 - 러스트(Rust)로 작성된 패리티티(Parity)
 - 고(Go)로 작성된 게스(Geth)
 - C++로 작성된 cpp-ethereum
 - 파이썬(Python)으로 작성된 pyethereum
 - 스칼라(Scala)로 작성된 맨티스(Mantis)
 - 자바(Java)로 작성된 하모니(Harmony)



이더리움 네트워크

- 이더리움 개발을 위한 네트워크 선택
 - 풀 노드
 - 공개 테스트넷
 - 로컬 블록체인 시뮬레이션



이더리움 네트워크

- 풀 노드의 장단점

- 장점

- 이더리움 기반 네트워크의 복원력과 검열 저항 지원
 - 모든 트랜잭션을 정식으로 검증
 - 중개자 없이 공개 블록체인의 모든 컨트랙트와 상호작용할 수 있음
 - 중개자 없이 컨트랙트를 공개 블록체인에 직접 배포할 수 있음
 - 블록체인 상태(계정, 스마트 컨트랙트 등)를 오프라인에서 조회할 수 있음
 - 읽은 정보를 제3자에게 노출하지 않고 가져올 수 있음

- 단점

- 하드웨어와 대역폭 자원의 확대가 필요
 - 처음 시작할 때 전체 동기화를 위해 여러 날이 소요
 - 동기화를 유지하기 위해 관리하고, 업그레이드하고, 온라인 상태로 유지해야 함



이더리움 네트워크

- 공개 테스트넷의 장단점

- 장점

- 테스트넷 노드는 훨씬 적은 데이터와 동기화를 필요로 함
 - 테스트넷 노드는 몇 시간 내에 전체 동기화를 할 수 있음
 - 컨트랙트 배포 및 트랜잭션 생성을 위한 테스트용 이더를 몇몇 Faucet으로부터 무료로 얻을 수 있음
 - 테스트넷은 다른 많은 스마트 컨트랙트가 동작 실행 중인 공개 블록체인임

- 단점

- 실제 돈을 사용하지 않고 테스트 이더로 실행하기 때문에 실전 보안성 테스트는 할 수 없음
 - 퍼블릭 블록체인에서만 실전 테스트를 할 수 없음



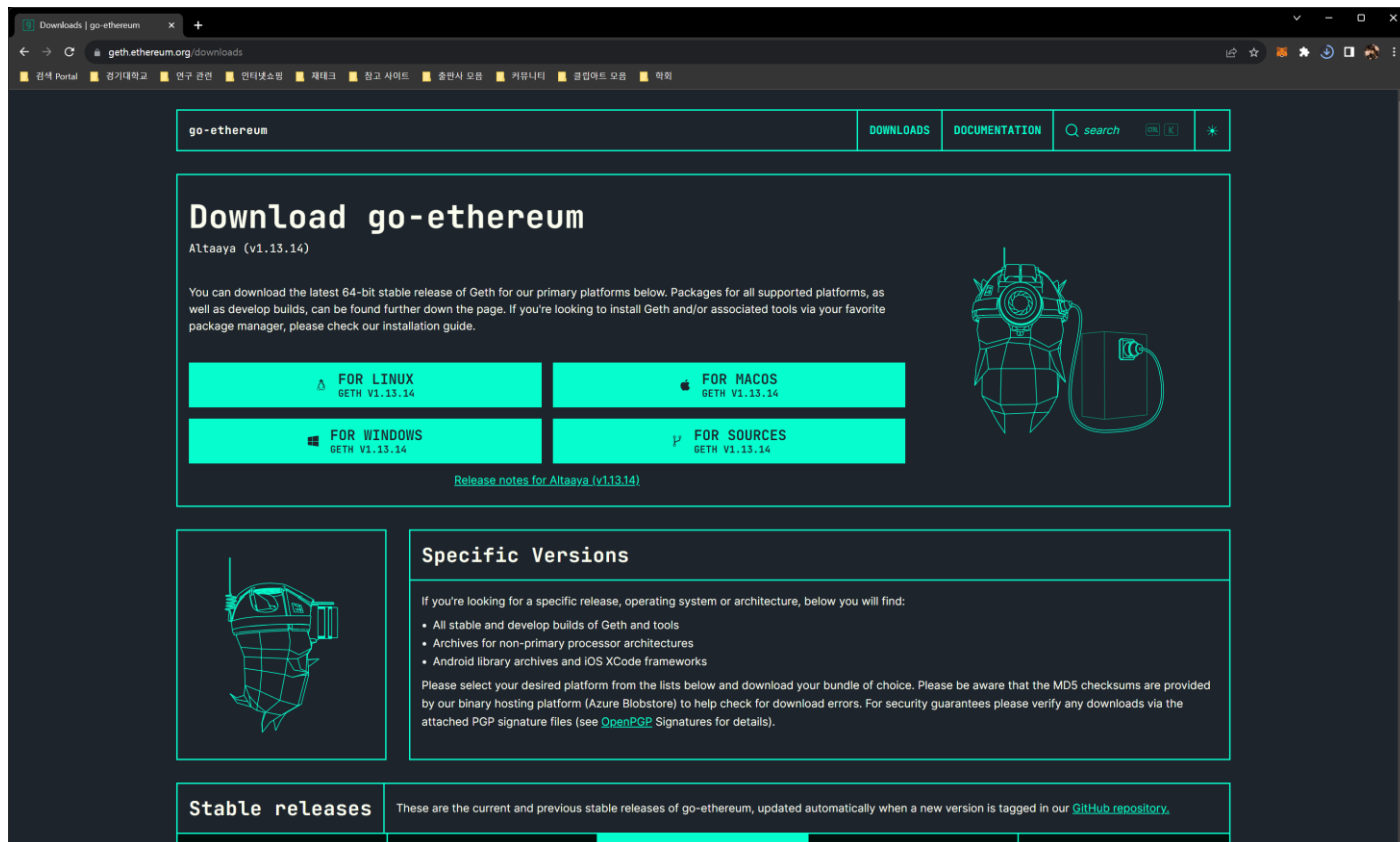
이더리움 네트워크

- 로컬 블록체인 시뮬레이션 장단점
 - 단일 인스턴스 사설 블록체인 (ex. 가나슈)
 - 장점
 - 동기화가 없고 디스크에 데이터가 거의 없음
 - 테스트 이더를 얻을 필요가 없음
 - 다른 사용자가 없음
 - 다른 컨트랙트 역시 없음
 - 단점
 - 공개 블록체인과 동일하게 동작하지 않음
 - 트랜잭션 순서나 공간을 두고 경쟁이 없음
 - 채굴이 예측 가능함
 - 테스트를 위해 의존성을 갖는 것들과 컨트랙트 라이브러리 등을 직접 배포해야 함



이더리움 클라이언트 실행

- 게스(geth, go-ethereum) 설치
 - <https://geth.ethereum.org/downloads>에서 운영체제에 맞는 설치 파일 이용

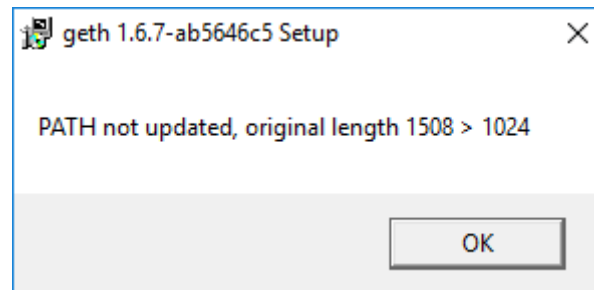


The screenshot shows the 'Downloads | go-ethereum' page in a web browser. The page has a dark theme with a blue header bar containing the 'go-ethereum' logo, 'DOWNLOADS', 'DOCUMENTATION', and a search bar. Below the header, the main section is titled 'Download go-ethereum' with the subtitle 'Altaaya (v1.13.14)'. It includes a paragraph explaining that users can download the latest 64-bit stable release for primary platforms (Linux, macOS, Windows, and Sources). To the right of this text is a large illustration of a robot head. Below the text are four blue buttons: 'FOR LINUX GETH V1.13.14', 'FOR MACOS GETH V1.13.14', 'FOR WINDOWS GETH V1.13.14', and 'FOR SOURCES GETH V1.13.14'. A link for 'Release notes for Altaaya (v1.13.14)' is provided. Further down, there is a section titled 'Specific Versions' with a list of links for stable and develop builds, non-primary processor architectures, and Android/iOS frameworks. A paragraph explains that users should select their desired platform and download their bundle of choice, noting that MD5 checksums are provided by the binary hosting platform (Azure Blobstore) and that security guarantees are verified via attached PGP signature files. At the bottom, there is a 'Stable releases' section with a link to the 'GitHub repository'.



이더리움 클라이언트 실행

- 게스(geeth, go-ethereum) 설치
 - “PATH not updated, original length...” 메시지가 나올 경우
 - PATH 환경변수에 geeth 실행 파일이 있는 폴더를 추가하면 됨
 - <https://zeliard.tistory.com/entry/geeth-path-not-updated-original-length-1024>





이더리움 클라이언트 실행

- 게스(eth, go-ethereum) 설치
 - 구동 가능한 상태인지 확인

```
명령 프롬프트
Microsoft Windows [Version 10.0.22621.3296]
(c) Microsoft Corporation. All rights reserved.

C:\Users\LeeJaeheung>geth version
Geth
Version: 1.13.14-stable
Git Commit: 2bd6bd01d2e8561dd7fc21b631f4a34ac16627a1
Git Commit Date: 20240227
Architecture: amd64
Go Version: go1.21.6
Operating System: windows
GOPATH=
GOROOT=

C:\Users\LeeJaeheung>
```



이더리움 기반 블록체인의 첫 번째 동기화

- 게스(geth, go-ethereum) 실행
 - geth

```
C:\Users\LeeJaeheung>geth
INFO [03-19|15:44:03.998] Starting Geth on Ethereum mainnet...
INFO [03-19|15:44:04.017] Bumping default cache on mainnet      provided=1024 updated=4096
INFO [03-19|15:44:04.019] Maximum peer count                    ETH=50 total=50
INFO [03-19|15:44:04.022] Set global gas cap                    cap=50,000,000
INFO [03-19|15:44:04.022] Initializing the KZG library          backend=gokzg
INFO [03-19|15:44:04.047] Allocated trie memory caches          clean=614.00MiB dirty=1024.00MiB
INFO [03-19|15:44:04.047] Defaulting to pebble as the backing database
INFO [03-19|15:44:04.047] Allocated cache and file handles      database=C:\Users\LeeJaeheung\AppData\Local\Ethereum\
geth\chaindata cache=2.00GiB handles=8192
INFO [03-19|15:44:04.093] Opened ancient database               database=C:\Users\LeeJaeheung\AppData\Local\Ethereum\
geth\chaindata\ancient\chain readonly=false
INFO [03-19|15:44:04.093] State schema set to default           scheme=hash
INFO [03-19|15:44:04.093] Initialising Ethereum protocol        network=1 dbversion=<nil>
INFO [03-19|15:44:04.099] Writing default main-net genesis block
INFO [03-19|15:44:04.314] Persisted trie from memory database    nodes=12356 size=1.79MiB time=47.5974ms gcnodes=0 gcs
size=0.00B gctime=0s livenodes=0 livesize=0.00B
INFO [03-19|15:44:04.341]
INFO [03-19|15:44:04.341] -----
INFO [03-19|15:44:04.341] Chain ID: 1 (mainnet)
INFO [03-19|15:44:04.341] Consensus: Beacon (proof-of-stake), merged from Ethash (proof-of-work)
INFO [03-19|15:44:04.341]
INFO [03-19|15:44:04.341] Pre-Merge hard forks (block based):
INFO [03-19|15:44:04.341]   - Homestead:                        #1150000 (https://github.com/ethereum/execution-specs/blob/main/network-upgrades/mainnet-upgrades/homestead.md)
INFO [03-19|15:44:04.341]   - DAO Fork:                        #1920000 (https://github.com/ethereum/execution-specs/blob/main/network-upgrades/mainnet-upgrades/dao-fork.md)
INFO [03-19|15:44:04.341]   - Tangerine Whistle (EIP 150):    #2463000 (https://github.com/ethereum/execution-specs/blob/main/network-upgrades/mainnet-upgrades/tangerine-whistle.md)
```



이더리움 기반 블록체인의 첫 번째 동기화

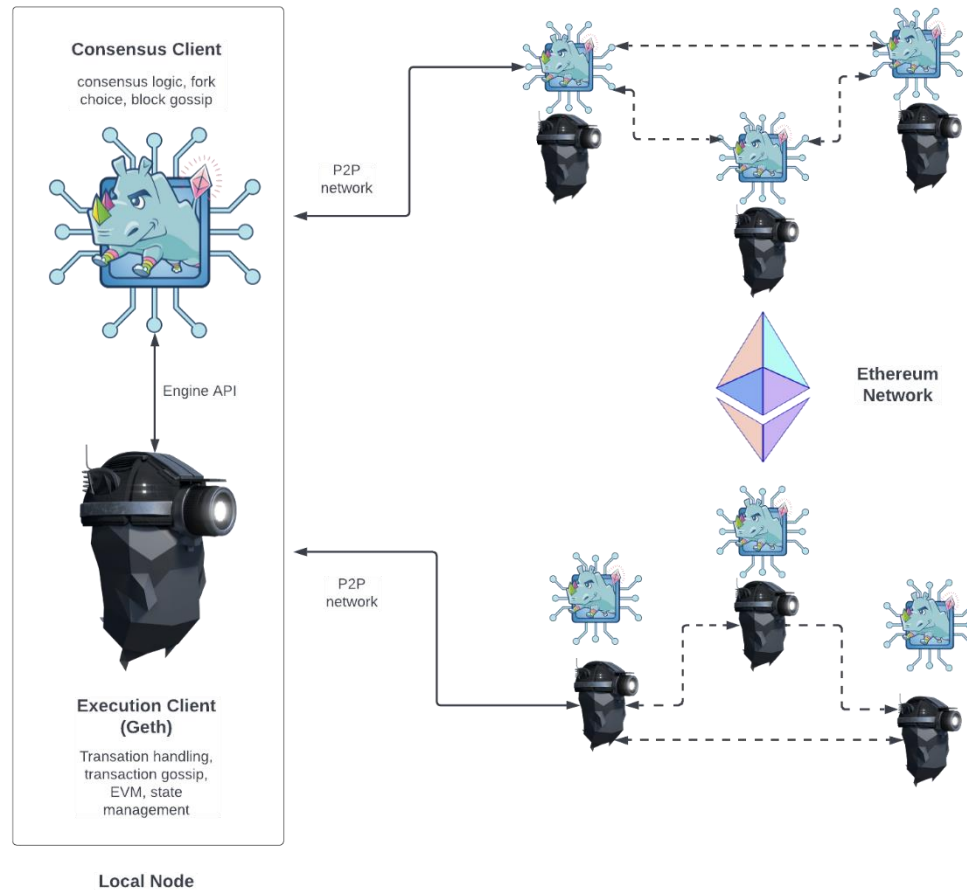
- 게스(eth, go-ethereum) 실행
 - 안되는 이유는? 합의 알고리즘의 변화!!!

```
명령 프롬프트 - geth
INFO [03-19|17:39:46.418] Gasprice oracle is ignoring threshold set threshold=2
WARN [03-19|17:39:46.421] Unclean shutdown detected booted=2024-03-19T17:31:11+0900 age=8m35s
WARN [03-19|17:39:46.421] Unclean shutdown detected booted=2024-03-19T17:31:53+0900 age=7m53s
WARN [03-19|17:39:46.422] Engine API enabled protocol=eth
INFO [03-19|17:39:46.422] Starting peer-to-peer node instance=Geth/v1.13.14-stable-2bd6bd01/windows-amd64/
go1.21.6
INFO [03-19|17:39:46.445] New local node record seq=1,710,830,644,474 id=c9956f7b60bbaf05 ip=127.0.0.
1 udp=30303 tcp=30303
INFO [03-19|17:39:46.446] Started P2P networking self=enode://aa11b69645ba2fe2d6676dd07df8159c354c5c34
3d5101c934a18ddfc9c12cfa33866fae8b94d4b94f4b45b564c42a1deb888cb8e2e99b209ad2c0692289d66d@127.0.0.1:30303
INFO [03-19|17:39:46.446] IPC endpoint opened url=\\.\pipe\geth.ipc
INFO [03-19|17:39:46.446] Loaded JWT secret file path=C:\Users\LeeJaeheung\AppData\Local\Ethereum\geth
\jwtsecret crc32=0x43cca87f
INFO [03-19|17:39:46.449] WebSocket enabled url=ws://127.0.0.1:8551
INFO [03-19|17:39:46.452] HTTP server started endpoint=127.0.0.1:8551 auth=true prefix= cors=localh
ost vhosts=localhost
INFO [03-19|17:39:48.696] New local node record seq=1,710,830,644,475 id=c9956f7b60bbaf05 ip=203.249.
8.112 udp=30303 tcp=30303
INFO [03-19|17:39:49.116] NAT mapped port proto=TCP extport=30303 intport=30303 interface="UPNP
IGDv2-IP1"
INFO [03-19|17:39:49.570] NAT mapped port proto=UDP extport=30303 intport=30303 interface="UPNP
IGDv2-IP1"
WARN [03-19|17:40:21.440] Post-merge network, but no beacon client seen. Please launch one to follow the chain!
WARN [03-19|17:45:21.735] Post-merge network, but no beacon client seen. Please launch one to follow the chain!
INFO [03-19|17:47:49.397] NAT mapped port proto=TCP extport=30303 intport=30303 interface="UPNP
IGDv2-IP1"
INFO [03-19|17:47:49.889] NAT mapped port proto=UDP extport=30303 intport=30303 interface="UPNP
IGDv2-IP1"
WARN [03-19|17:50:22.114] Post-merge network, but no beacon client seen. Please launch one to follow the chain!
```



이더리움 기반 블록체인의 첫 번째 동기화

- 이더리움 노드 아키텍처
 - 실행 클라이언트와 컨센서스 클라이언트





이더리움 기반 블록체인의 첫 번째 동기화

- 이더리움 노드 아키텍처
 - 실행 클라이언트
 - 이더리움 가상 머신을 실행시키는 역할 수행
 - 노드가 트랜잭션을 받으면 실행 클라이언트가 해당 트랜잭션을 실행하고, 그 결과를 블록으로 생성
 - 대표적인 실행 클라이언트
 - Geth (Go로 작성)
 - Nethermind (C#, .NET으로 작성)
 - Besu (Java로 작성)
 - Erigon (Go로 작성)
 - Akula (Rust로 작성)



이더리움 기반 블록체인의 첫 번째 동기화

- 이더리움 노드 아키텍처
 - 컨센서스 클라이언트
 - 분산 시스템에서 노드들이 합의를 이루는데 필요한 역할 수행
 - 이더리움은 PoW(Proof of Work)와 PoS(Proof of Stake) 두 가지 방식을 지원하는데, 컨센서스 클라이언트는 이러한 알고리즘을 구현하고 블록의 유효성을 검증하는 역할을 수행
 - 대표적인 컨센서스 클라이언트
 - Lighthouse (Rust로 작성)
 - Nimbus (Nim으로 작성)
 - Prysm (Go로 작성)
 - Teku (Java로 작성)
 - Lodestar (TypeScript로 작성)



이더리움 기반 블록체인의 첫 번째 동기화

- 이더리움 노드 아키텍처
 - 클라이언트 분포도





원격 이더리움 클라이언트

- 원격 클라이언트
 - 풀 클라이언트의 일부 기능을 제공
 - 더 빠르며 데이터 용량도 훨씬 더 적게 요구
 - 다음 기능 중 하나 이상을 제공
 - 개인 키와 이더리움 주소를 지갑에서 관리
 - 트랜잭션 생성, 서명 및 브로드캐스트
 - 데이터 페이로드(payload)를 사용하여 스마트 컨트랙트와 상호연동
 - 브라우저와 댕(DApp) 간 상호연동
 - 블록 탐색기 같은 외부 서비스 링크
 - 이더 단위를 변환하고 외부 소스에서 환율을 검색
 - 자바스크립트 객체로서 web3 인스턴스(instance)를 웹 브라우저에 삽입
 - 다른 클라이언트가 브라우저에 제공/삽입한 web3 인스턴스를 사용
 - 로컬 또는 원격 이더리움 노드에서 RPC 서비스로 접근



원격 이더리움 클라이언트

- 모바일(스마트폰) 지갑
 - 스테터스(Status, <https://status.app>)
 - 오픈소스 기반의 이더리움 모바일 월렛
 - 프라이빗 메신저, 보안 암호화폐 지갑, 이더리움 web3 디앱 브라우저를 결합
 - 트러스트월렛(Trust Wallet, <https://trustwallet.com/ko>)
 - 세계적인 암호화폐 거래소인 바이낸스(Binance)가 제공
 - 주요 기능
 - 암호화폐 구매, 교환 및 판매를 위한 내장 교환 기능
 - 사용자 지정 토큰을 포함하여 수천 개의 디지털 통화 저장 지원
 - 12개 이상의 암호화폐 자산 스테이킹을 위한 인앱 지원
 - 탈중앙화 애플리케이션 및 DeFi 브라우저



원격 이더리움 클라이언트

- 브라우저 지갑
 - 메타마스크 (MetaMask, <https://metamask.io/>)
 - 다재다능한 브라우저 기반 지갑
 - RPC 클라이언트 및 컨트랙트 탐색기 기능도 지원
 - 다양한 이더리움 블록체인에 연결되는 RPC 클라이언트 역할을 하는 브라우저 자바스크립트 컨텍스트에 web3 인스턴스를 넣음
 - 마이이더월렛 (MyEtherWallet, <https://www.myetherwallet.com/>)
 - 브라우저 기반 자바스크립트 원격 클라이언트
 - 테스트와 하드웨어 지갑을 위한 인터페이스로서 매우 유용
 - 마이크립토 (MyCrypto, <https://www.mycrypto.com/>)
 - 2018년 2월 테일러 모나한 (Taylor Monahan)이 마이이더월렛 팀 대부분과 함께 마이이더월렛을 포크하여 만든 암호화폐 지갑

