CHAPTER3 스마트 컨트랙트 이론

What is Smart Contract

What is smart contract? (ref #12)

Smart contract

A set of promises specified in digital form,
including protocols within which
the parties perform on these promises.

Why

Observability

Verifiability

Privity

Enforceability

Smart Contracts: Building Blocks for Digital Markets

Glossary

(This is a partial rewrite of the article which appeared in Extropy #16)

Introduction

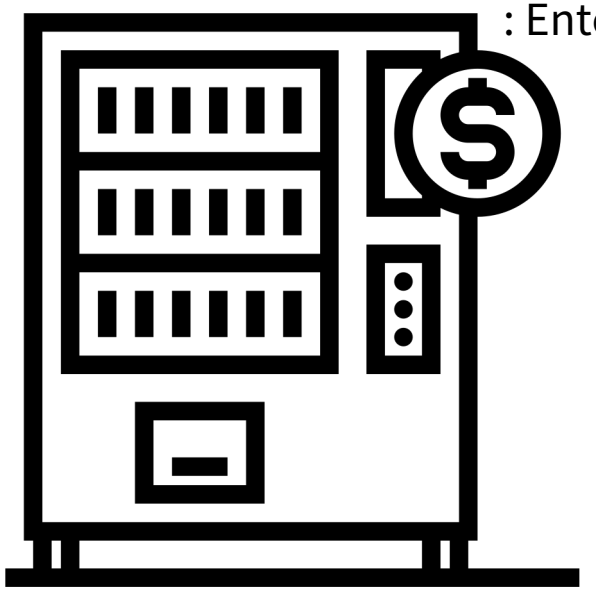
The contract, a set of promises agreed to in a "meeting of the minds", is the traditional way to formalize a relationship. While contracts are primarily used in business relationships (the focus of this article), they can also involve personal relationships such as marraiges. Contracts are also important in politics, not only because of "social contract" theories but also because contract enforcement has traditionally been considered a basic function of capitalist governments.

Whether enforced by a government, or otherwise, the contract is the basic building block of a free market economy. Over many centuries of cultural evolution has emerged both the concept of contract and principles related to it, encoded into common law. Algorithmic information theory suggests that such evolved structures are often prohibitively costly to recompute. If we started from scratch, using reason and experience, it could take many centuries to redevelop sophisticated ideas like property rights that make the modern free market work [Hayek].

The success of the common law of contracts, combined with the high cost of replacing it, makes it worthwhile to both preserve and to make use of these principles where appropriate. Yet, the digital revolution is radically changing the kinds of relationships we can have. What parts of our hard-won legal tradition will still be valuable in the cyberspace era? What is the best way to apply these common law principles to the design of our on-line relationships?

Computers make possible the running of algorithms heretofore prohibitively costly, and networks the quicker

What is smart contract? (ref #13)



A defined input
: Entering a coin

A consequential action
: The transfer of ownership

An underlying asset

The Idea of Smart Contracts

What is the meaning and purpose of "security"? How does it relate the the relationships we have? I argue that the formalizations of our relationships -- especially contracts -- provide the blueprint for ideal security.

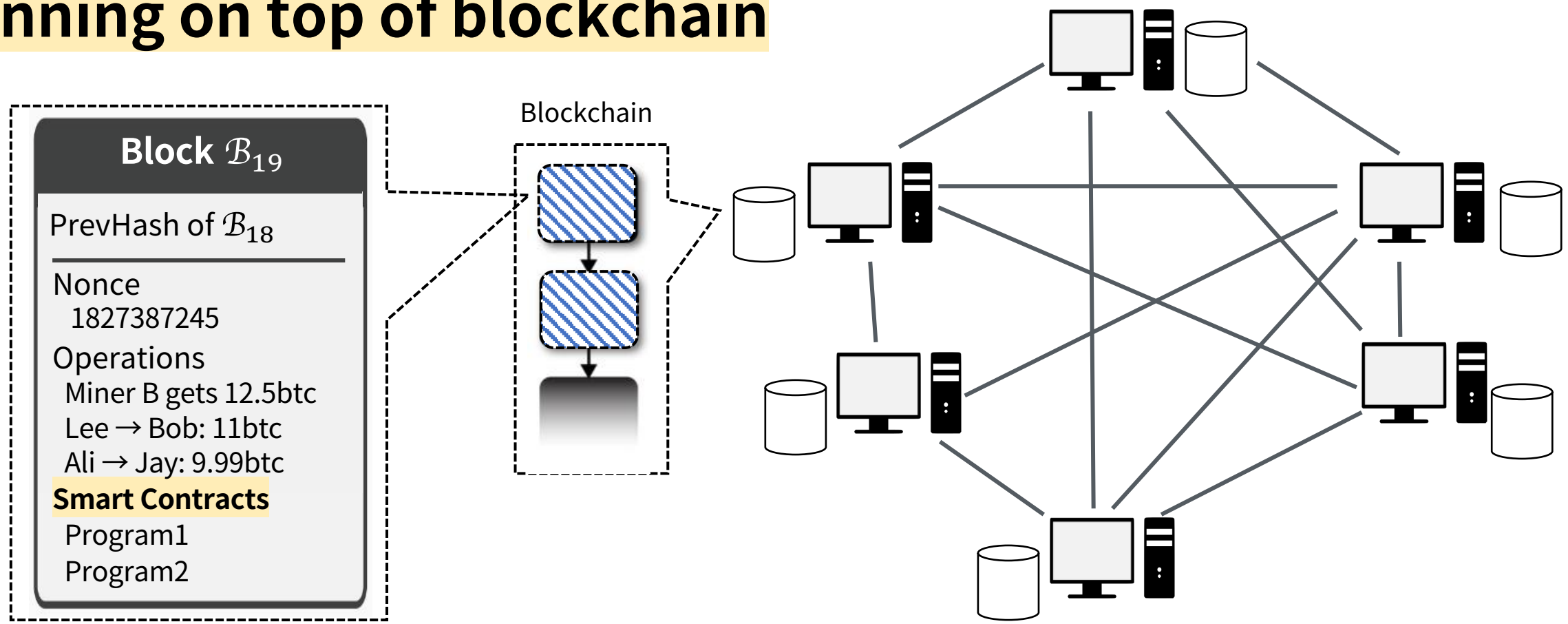
Many kinds of contractual clauses (such as collateral, bonding, delineation of property rights, etc.) can be embedded in the hardware and software we deal with, in such a way as to make breach of contract expensive (if desired, sometimes prohibitively so) for the breacher. A canonical real-life example, which we might consider to be the primitive ancestor of smart contracts, is the humble vending machine. Within a limited amount of potential loss (the amount in the till should be less than the cost of breaching the mechanism), the machine takes in coins, and via a simple mechanism, which makes a freshman computer science problem in design with finite automata, dispense change and product according to the displayed price. The vending machine is a contract with bearer: anybody with coins can participate in an exchange with the vendor. The lockbox and other security mechanisms protect the stored coins and contents from attackers, sufficiently to allow profitable deployment of vending machines in a wide variety of areas.

Smart contracts go beyond the vending machine in proposing to embed contracts in all sorts of property that is valuable and controlled by digital means. Smart contracts reference that property in a dynamic, often proactively enforced form, and provide much better observation and verification where proactive measures must fall short.

What is smart contract?

A computer program

running on top of blockchain



What is smart contract?

A computer program

running on top of blockchain

Replicated (Across distributed nodes)

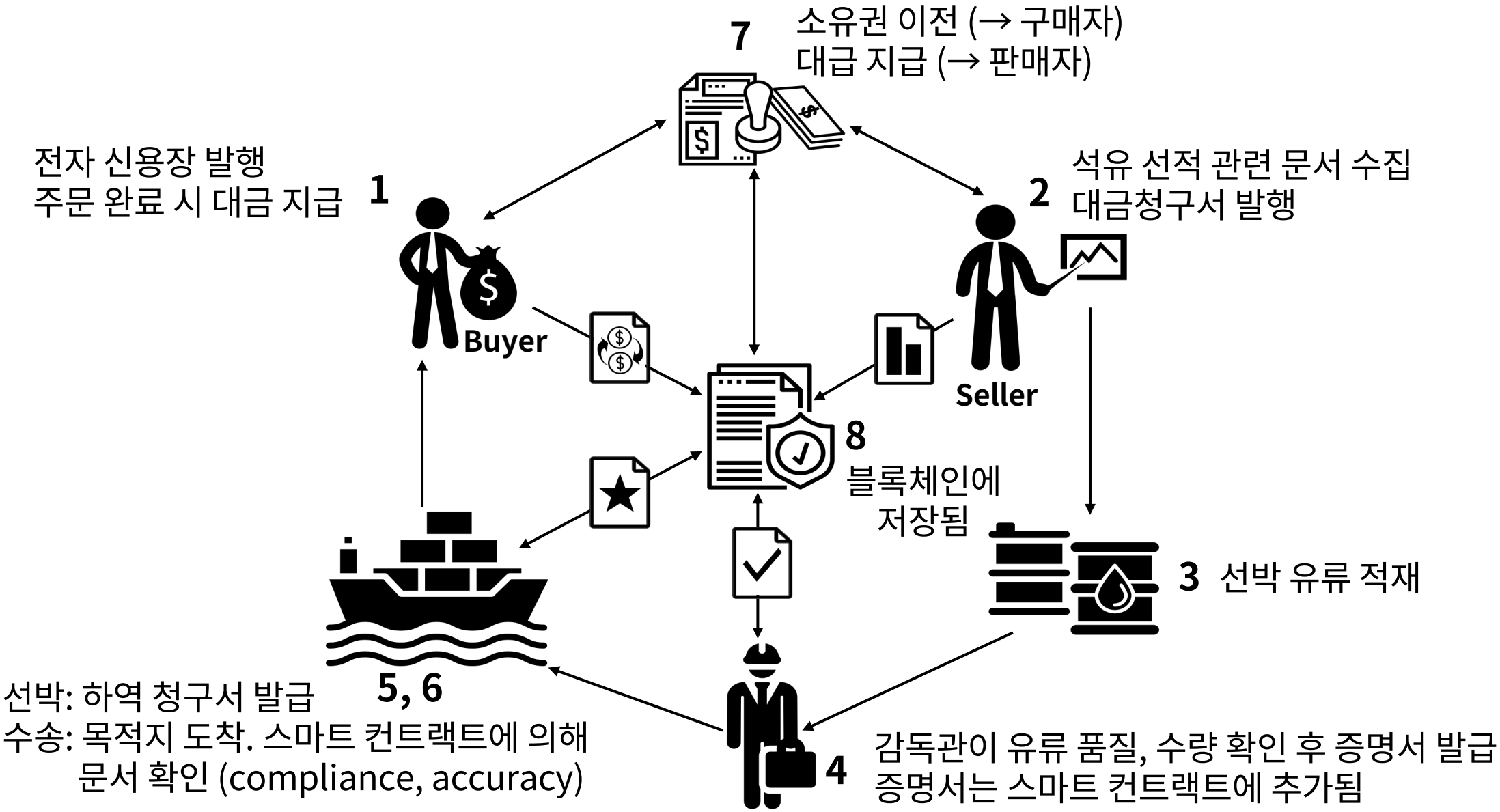
Immutable (Transparent)

Deterministic (The same for everyone)

Neutral & Passive (A bunch of codes)

Automated (Executed by nodes)

Smart Contracts in Trade Finance (WSJ)



Smart Contracts in Account-based Model

Second Used Contract in Block 5

assets = [(12, Sue), (34, Lee), ⋯]

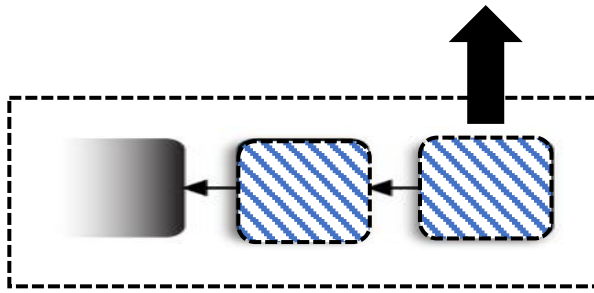
sellCar(price) { IF A THEN B }

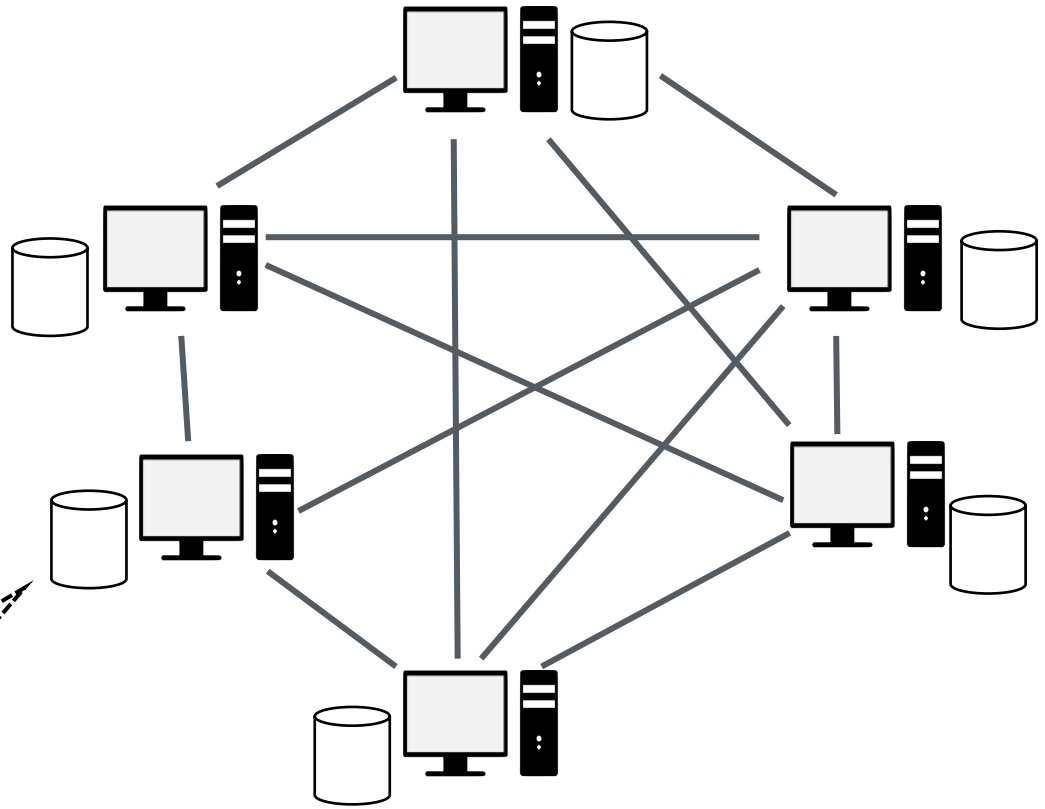
buyCar(carID, price) { IF C THEN D }

changeOwner(carID, newOwner) {
A new Owner owns a car of carID
}

State

Alice: {
Balance: 6.1,
Counter: 12,
…
},
carCompany: {
Balance: 100,
Contracts: [
CustomerSale,
SecondUsed
]
},
…





Smart Contracts in Account-based Model

State

```
Alice: {
  Balance: 6.1,
  Counter: 12,
  …
},
carCompany: {
  Balance: 100,
  Contracts: [
    CustomerSale,
    SecondUsed
  ]
},
…
```

Second Used Contract in Block 5

assets = [(12, Sue), (34, **Lee**), …]

sellCar(price) { IF A THEN B }

buyCar(carID, price) { IF C THEN D }

changeOwner(carID, newOwner) {
  A new Owner owns a car of carID
}

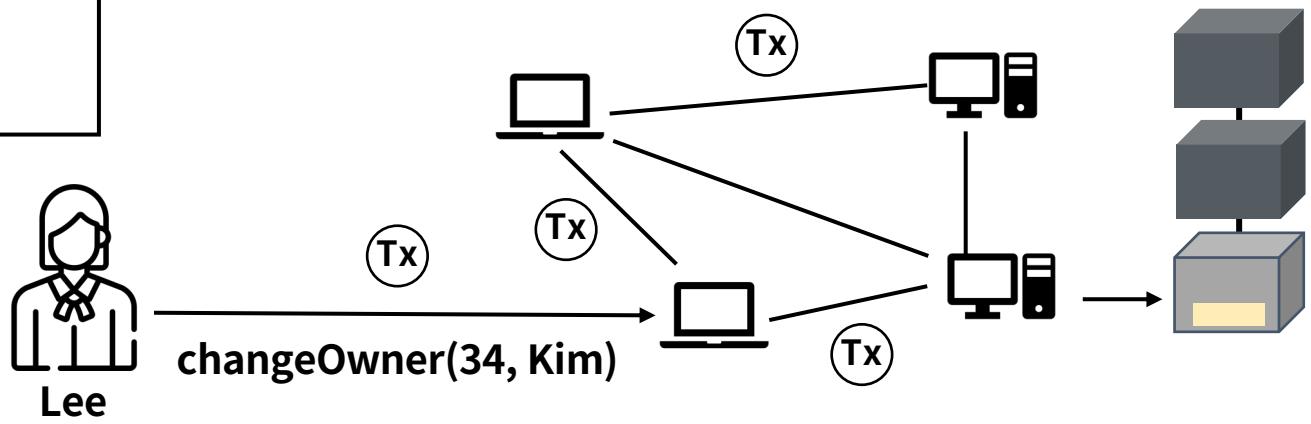
Second Used Contract in Block 9

assets = [(12, Sue), (34, **Kim**), …]

sellCar(price) { IF A THEN B }

buyCar(carID, price) { IF C THEN D }

changeOwner(carID, newOwner) {
  A new Owner owns a car of carID
}



From: Lee
To: Second Used Conract
Func: changeOwner
Params: [34, Kim]

Smart Contracts in Car Trading

Car Contract

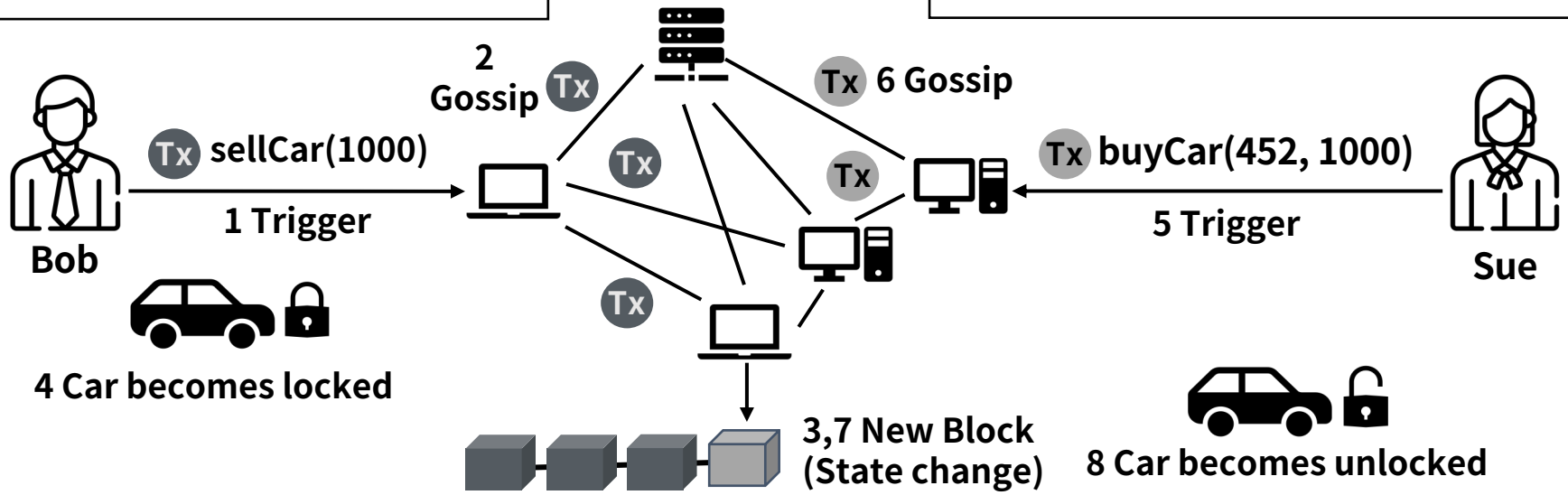
assets = [(123, Alice), (452, **Bob**), ⋯]

```
sellCar(price) {
  IF (msgSender has a car)
  THEN (Lock this car)
}

buyCar(carNum, price) {
  IF (price is equal to carNum.price)
  THEN (msgSender owns this car)
}
```

Car Contract

assets = [(123, Alice), (452, **Sue**), ⋯]

```
sellCar(price) {
  IF (msgSender has a car)
  THEN (Lock this car)
}

buyCar(carNum, price) {
  IF (price is equal to carNum.price)
  THEN (msgSender owns this car)
}
```



Tezos SW Architecture

Tezos Network*

Mainnet (471 Bakers / 9,159 Nodes)

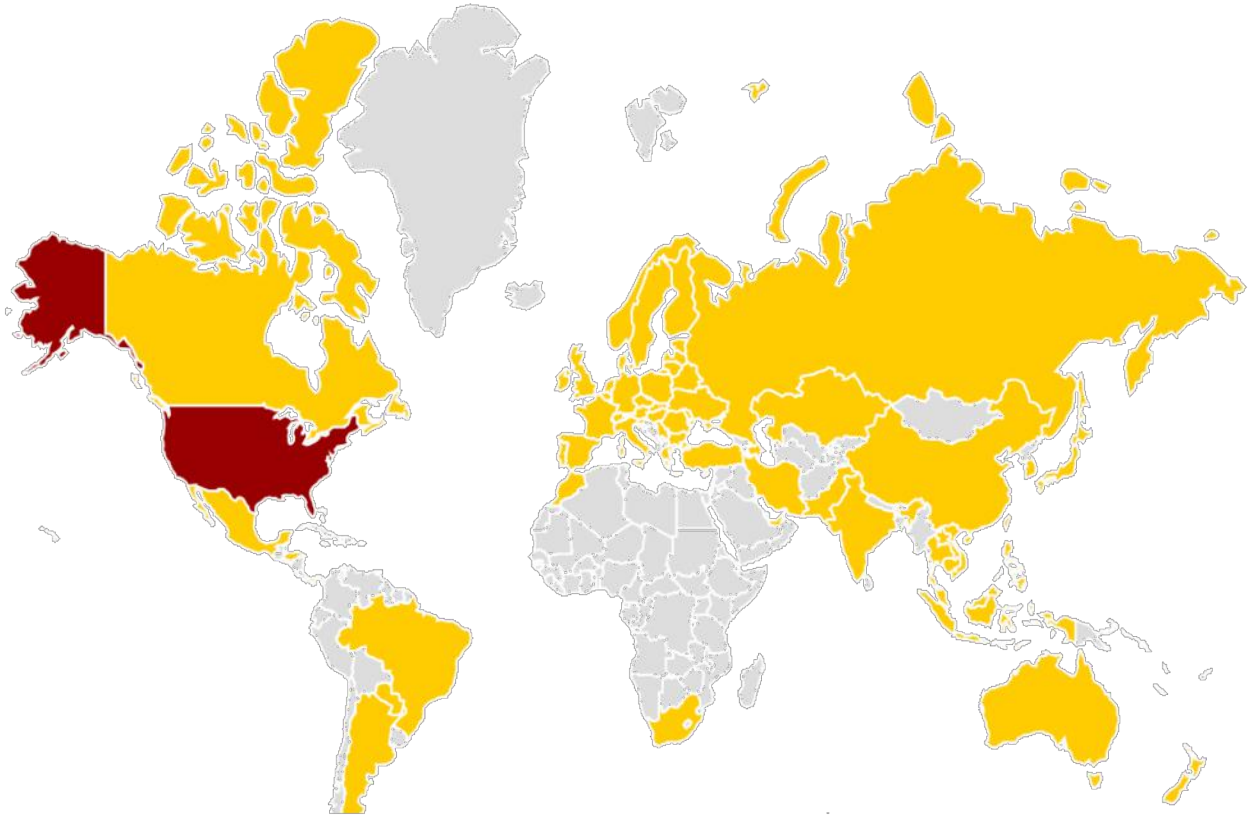
- Live network
- Production system
- Real Tezzies

Alphanet (19 Bakers / 2,294 Nodes)

- Test network
- Support all updates (staging)
- Fake Tezzies

Zeronet (12 Bakers / 114 Nodes)

- Test network
- Smaller (Cycle, intervals, …)
- For core developers



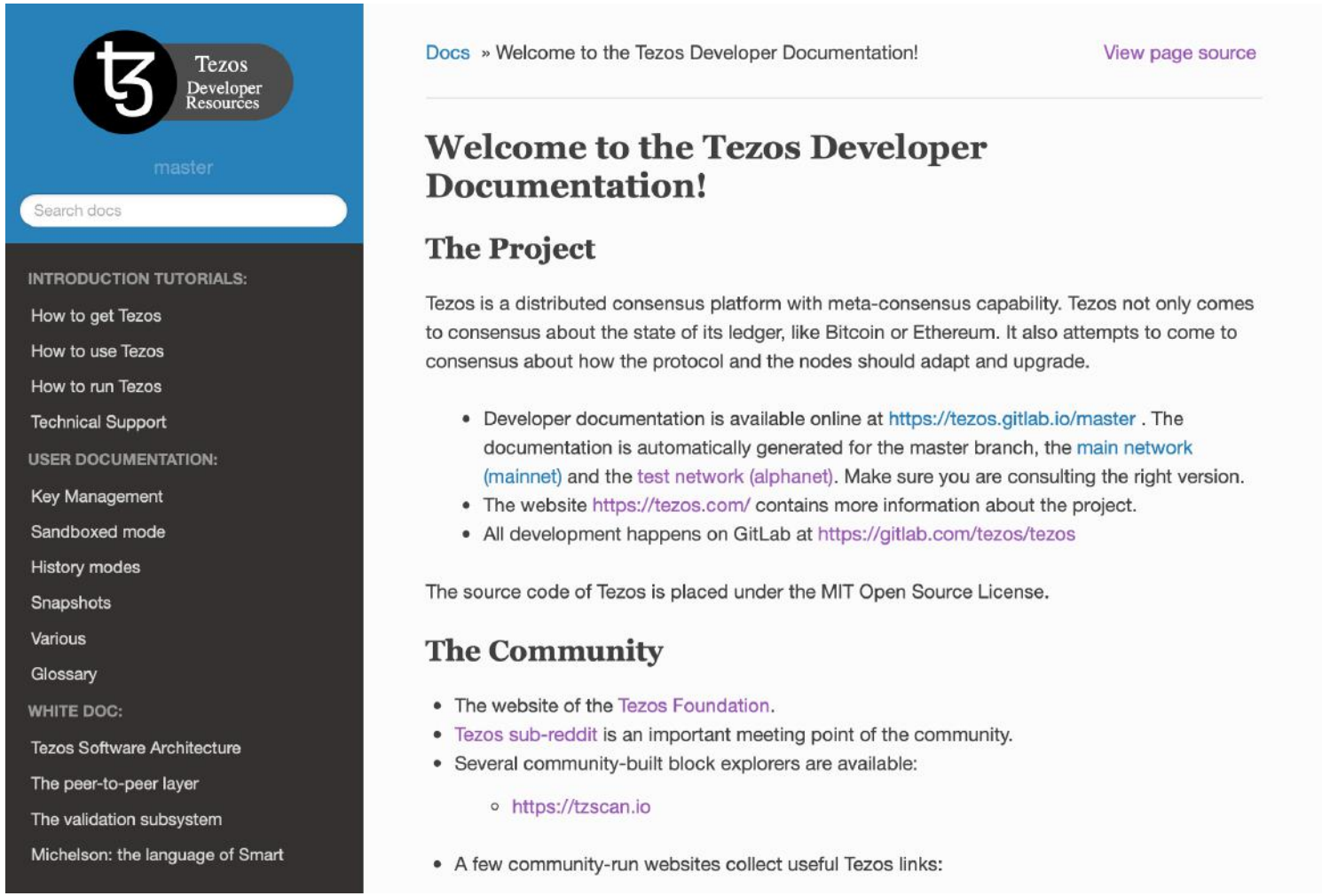
* As of 19.06.17

Running Alphanet (ref #14)

알파넷 설치

Docker Image

Installing from source



Running Alphanet

Installing Tezos SW from source

Being developed for Linux x86_64, mostly for Debian/Ubuntu and Archlinux.

Currently reported to work at:

- macOS/x86_64
- Linux/armv7h (32bit) (Raspberry Pi3, etc.)
- Linux/aarch64 (64bit) (Raspberry Pi3, etc.)

Hardware specification

- RAM: 4GB (8GB recommended)
- Storage: 20~40GB (SSD is strongly recommended)
- Good internet connection

Running Alphanet

Installing commands

```
$ sudo apt update
$ sudo apt install -y rsync git make m4 build-essential patch unzip wget
$ wget http://security.ubuntu.com/ubuntu/pool/main/b/bubblewrap/bubblewrap_0.2.1-1ubuntu0.1_amd64.deb
$ sudo dpkg -i bubblewrap_0.2.1-1ubuntu0.1_amd64.deb
$ wget https://github.com/ocaml/opam/releases/download/2.0.3/opam-2.0.3-x86_64-linux
$ sudo cp opam-2.0.3-x86_64-linux /usr/local/bin/opam
$ sudo chmod a+x /usr/local/bin/opam
$ git clone https://gitlab.com/tezos/tezos.git
$ cd tezos
$ git checkout alphanet
$ export OPAMNO=true
$ opam init --bare
$ export OPAMNO=false
$ sudo apt install -y libev-dev libgmp-dev pkg-config libhidapi-dev
$ make build-deps
$ eval $(opam env)
$ make
$ export PATH=~/tezos:$PATH
$ source ./src/bin_client/bash-completion.sh
$ export TEZOS_CLIENT_UNSAFE_DISABLE_DISCLAIMER=Y
```

알파넷 설치 환경 준비

아래 옵션 중 선택

1) MacOS 또는 Linux (Ubuntu 권장)가 준비되어 있는 경우 해당 PC 사용

2) **Ram 8기가 이상**의 윈도우 PC일 경우 Virtual Box 설치 후 **우분투 18.04** 설치

3) 클라우드 컴퓨팅 서비스 이용 (ex. 구글 클라우드)

구글 클라우드 가입

https://console.cloud.google.com/

구글 클라우드

구글 클라우드 가입 (https://console.cloud.google.com/)

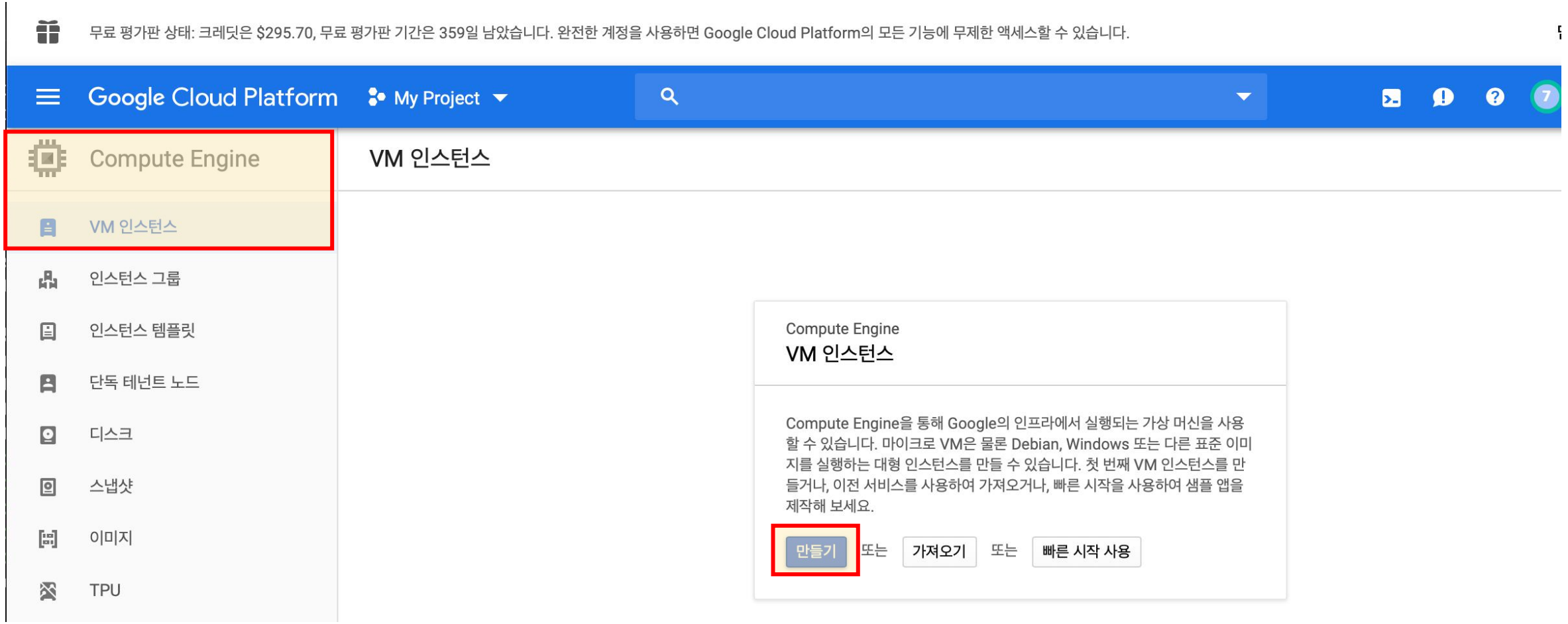
VM 인스턴스 생성 (최초 가입 시 300달러 크레딧 지급, 카드 정보 기입 필수)

카드 정보를 입력한 경우, 상단에 남아있는 무료 크레딧이 표기됨. 사용하지 않을 경우 인스턴스 삭제!

구글 클라우드

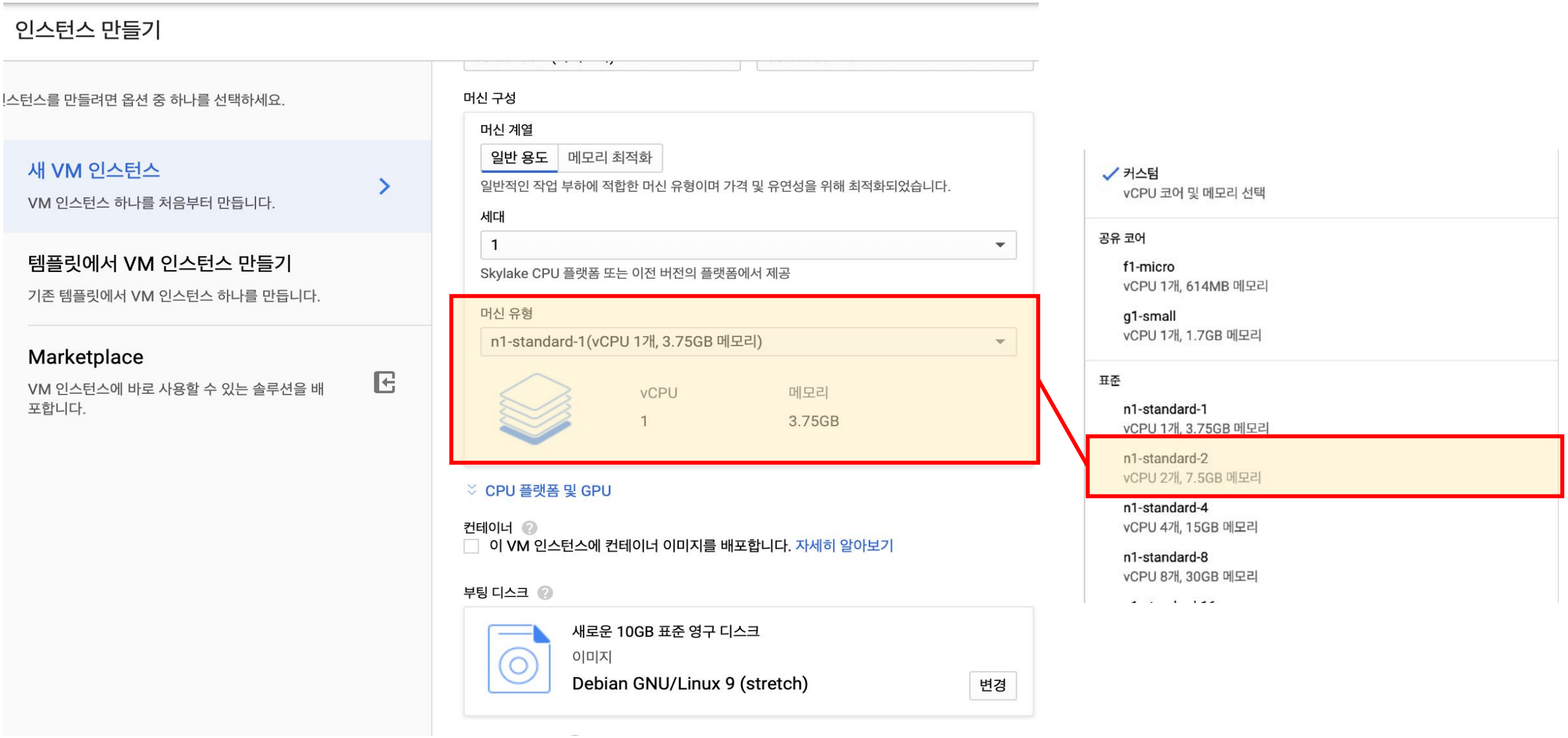
‘Compute Engine, VM 인스턴스’ 확인

만들기 클릭 (카드 정보는 이 단계에서 입력)



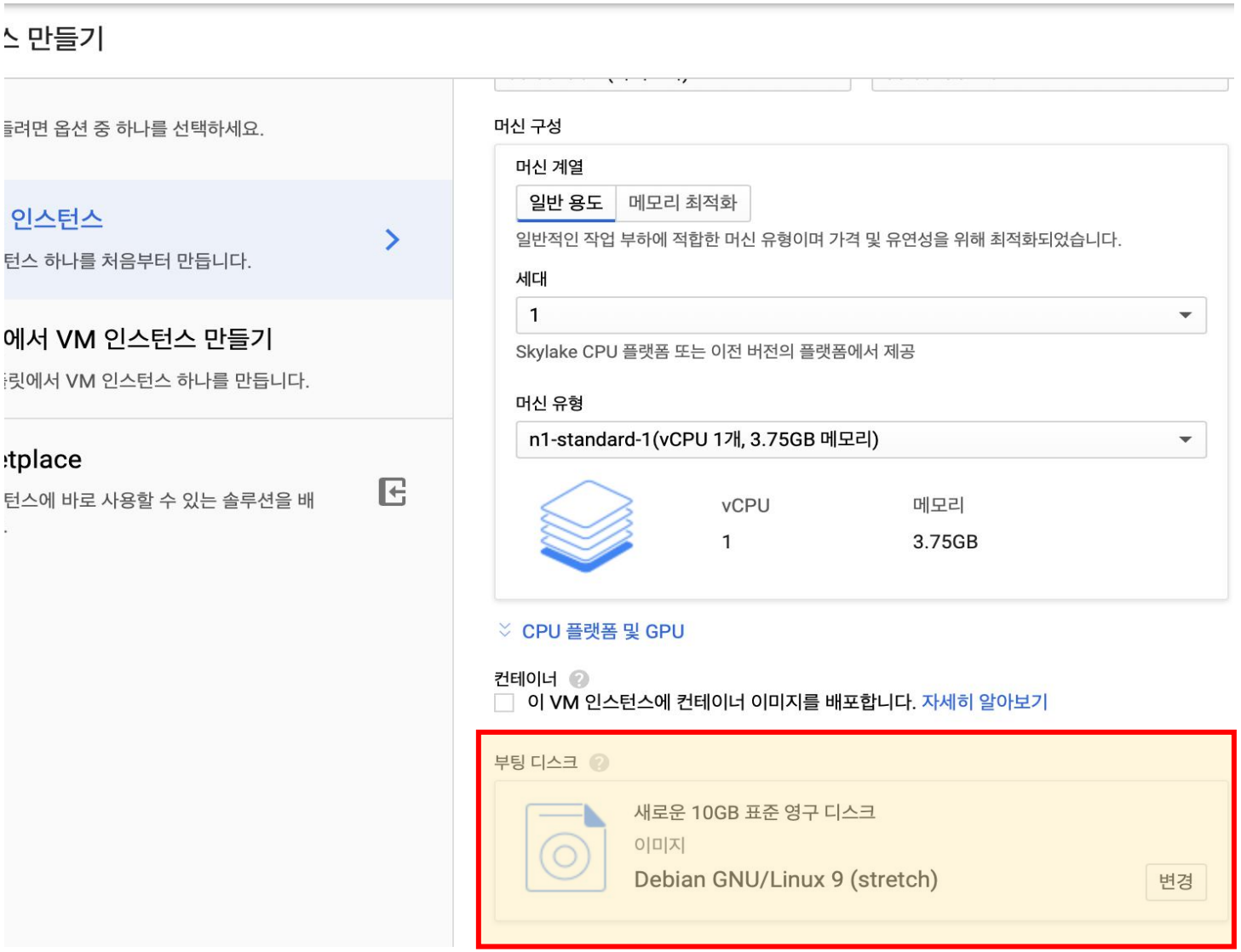
구글 클라우드 (머신 유형)

머신 유형과 **부팅 디스크** 설정 변경

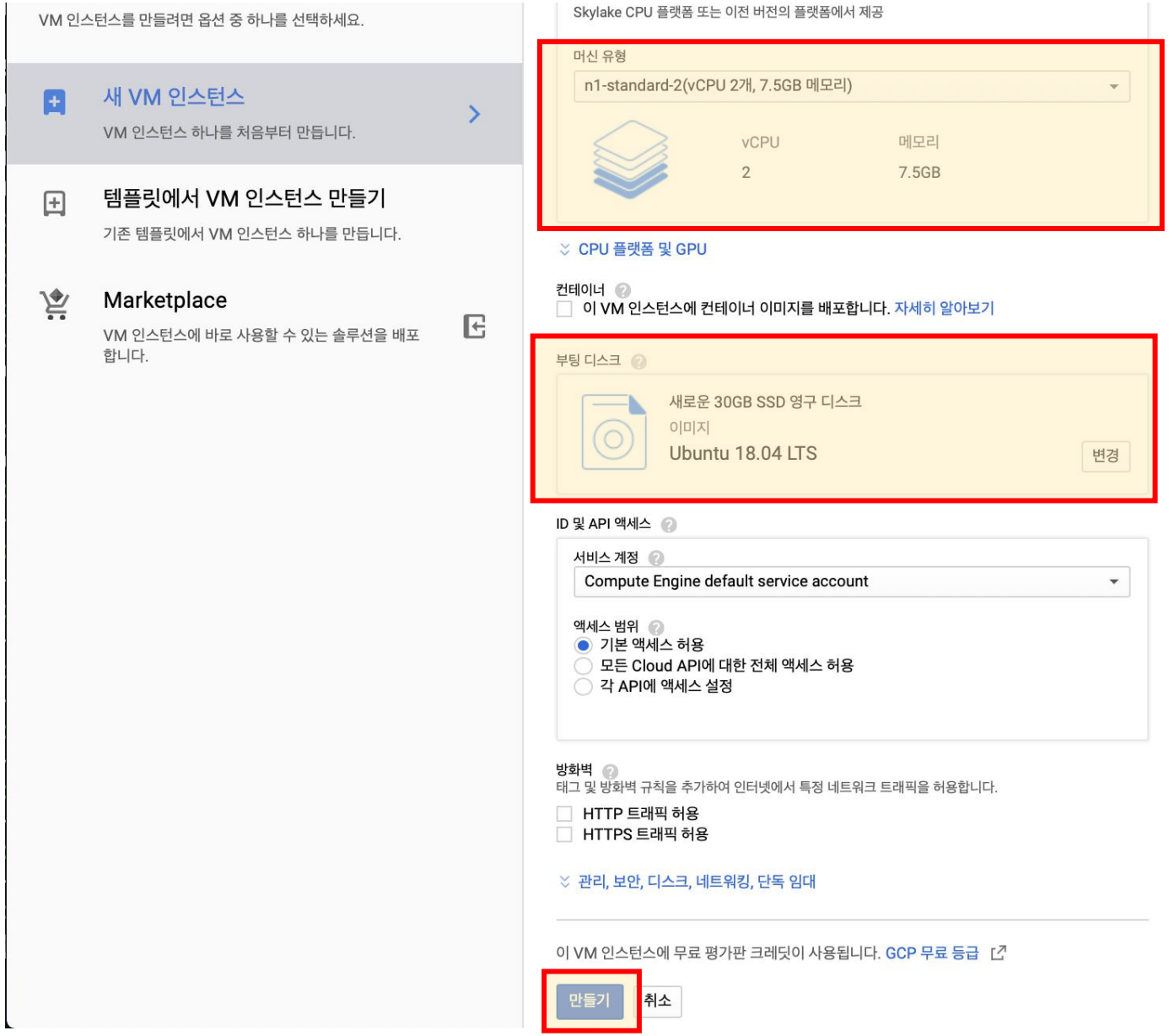


구글 클라우드 (부팅 디스크)

머신 유형과 **부팅 디스크** 설정 변경



구글 클라우드



머신 유형과 **부팅 디스크** 설정 확인 후 **만들기**

이 후 슬라이드 안내 따라서 Tezos SW 설치, 스냅샷 불러오기(Import)

Running Alphanet (ref #15)

Shortcut (Automated script)

All commands are packed together into a bash script.

It will install dependencies, download the source code, and build the code called “make process”.

```
$ curl "https://gitlab.com/tezoskorea/quickstart/raw/master/tz_install.sh" | bash -s alphanet
```

For macOS, use this

```
$ curl "https://gitlab.com/tezoskorea/quickstart/raw/master/tz_install_mac.sh" | bash -s alphanet
```

It takes 10 ~ 15 minutes.

```
Info: creating file vendors/ocplib-resto/lib_resto-directory/dune-project with this contents:
| (lang dune 1.8)
| (name ocplib-resto-directory)

Info: creating file vendors/ocplib-resto/lib_resto-json/dune-project with this contents:
| (lang dune 1.8)
| (name ocplib-resto-json)

Finished!
ubuntu@ip-172-31-18-57:~$
```

Running Alphanet

Binaries (Executable files)

```
ubuntu@ip-172-31-23-117:~/tezos$ ls -l
total 253932
-rw-rw-r--  1 ubuntu ubuntu     1114 Jun 24 12:32 LICENSE
-rw-rw-r--  1 ubuntu ubuntu     5031 Jun 24 12:32 Makefile
-rw-rw-r--  1 ubuntu ubuntu     1692 Jun 24 12:32 README.md
drwxrwxr-x  6 ubuntu ubuntu     4096 Jun 24 12:53 _build
drwxr-xr-x  9 ubuntu ubuntu     4096 Jun 24 12:42 _opam
-rw-rw-r--  1 ubuntu ubuntu       13 Jun 24 12:32 active_protocol_versions
-rw-rw-r--  1 ubuntu ubuntu     2419 Jun 24 12:32 contributing.md
drwxrwxr-x 12 ubuntu ubuntu     4096 Jun 24 12:32 docs
-rw-rw-r--  1 ubuntu ubuntu      193 Jun 24 12:32 dune
-rw-rw-r--  1 ubuntu ubuntu       16 Jun 24 12:53 dune-project
-rw-rw-r--  1 ubuntu ubuntu       16 Jun 24 12:32 dune-workspace
drwxrwxr-x  2 ubuntu ubuntu     4096 Jun 24 12:32 emacs
drwxrwxr-x  5 ubuntu ubuntu     4096 Jun 24 12:32 scripts
drwxrwxr-x 36 ubuntu ubuntu     4096 Jun 24 12:32 src
drwxrwxr-x  9 ubuntu ubuntu     4096 Jun 24 12:32 tests_python
-rwxrwxr-x  1 ubuntu ubuntu 31047200 Jun 24 12:55 tezos-accuser-004-Pt24m4xi
-rwxrwxr-x  1 ubuntu ubuntu 36000992 Jun 24 12:55 tezos-admin-client
-rwxrwxr-x  1 ubuntu ubuntu 31047168 Jun 24 12:55 tezos-baker-004-Pt24m4xi
-rwxrwxr-x  1 ubuntu ubuntu 36019608 Jun 24 12:55 tezos-client
-rwxrwxr-x  1 ubuntu ubuntu 31047216 Jun 24 12:55 tezos-endorser-004-Pt24m4xi
-rwxrwxr-x  1 ubuntu ubuntu 49637424 Jun 24 12:55 tezos-node
-rwxrwxr-x  1 ubuntu ubuntu 26499448 Jun 24 12:55 tezos-protocol-compiler
-rwxrwxr-x  1 ubuntu ubuntu 18645920 Jun 24 12:55 tezos-signer
drwxrwxr-x 13 ubuntu ubuntu     4096 Jun 24 12:32 vendors
```

Tezos SW Components

tezos-node
Network shell, The Tezos daemon

tezos-client
A command-line client and basic wallet

tezos-admin-client
Administration tool for the node

tezos-{baker, endorser, accuser}-004-Pt24m4xi
Daemons to bake, endorse and accuse

tezos-signer
A client to remotely sign operations or blocks

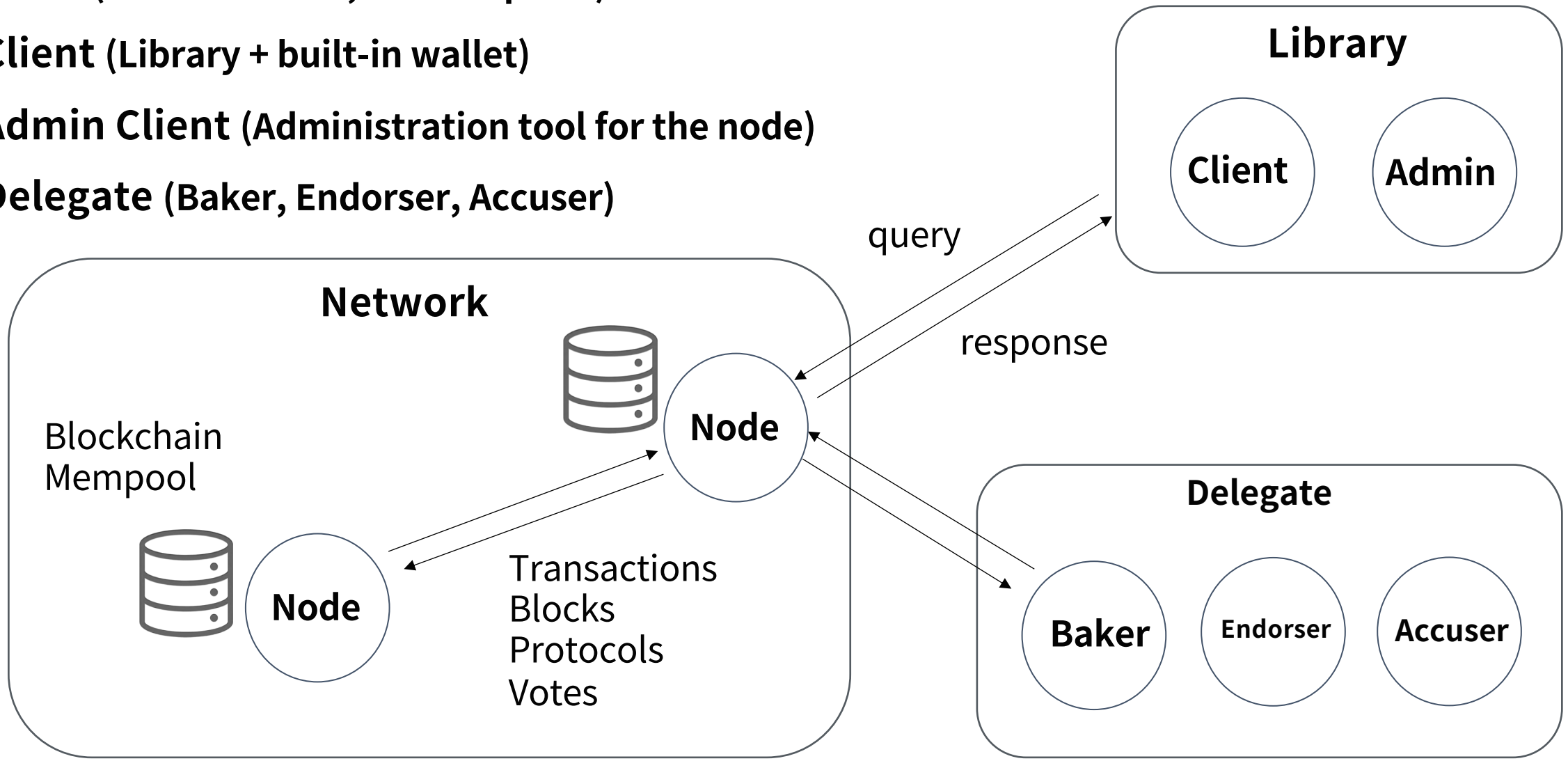
Tezos SW Components

Node (Network Shell, API end point)

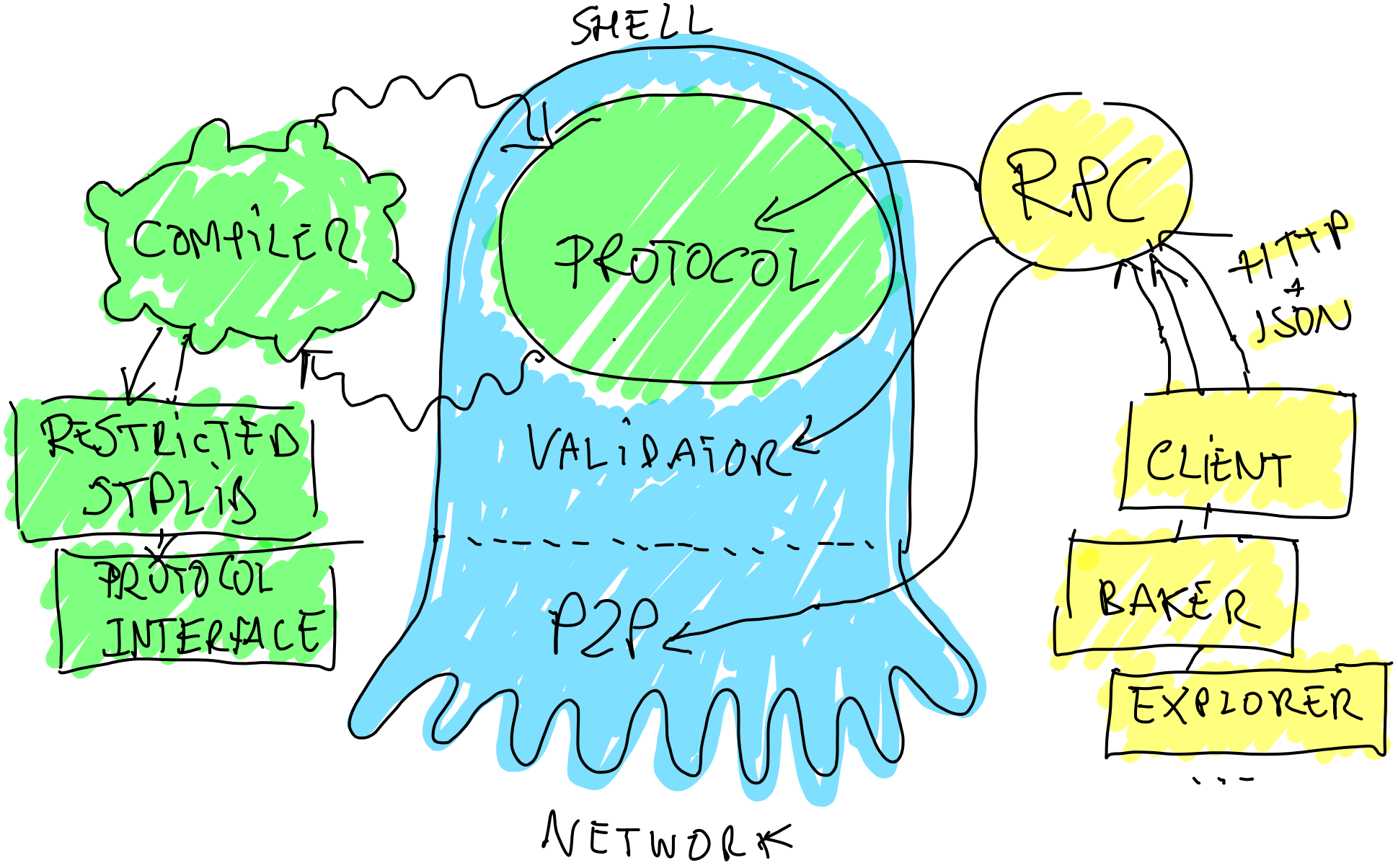
Client (Library + built-in wallet)

Admin Client (Administration tool for the node)

Delegate (Baker, Endorser, Accuser)



Tezos SW Architecture (ref #16)



Blockchain Protocols

Network protocol

Transactions / Blocks download and broadcast

P2P network management (discover, add, remove)

Relatively uncontroversial

Transaction protocol

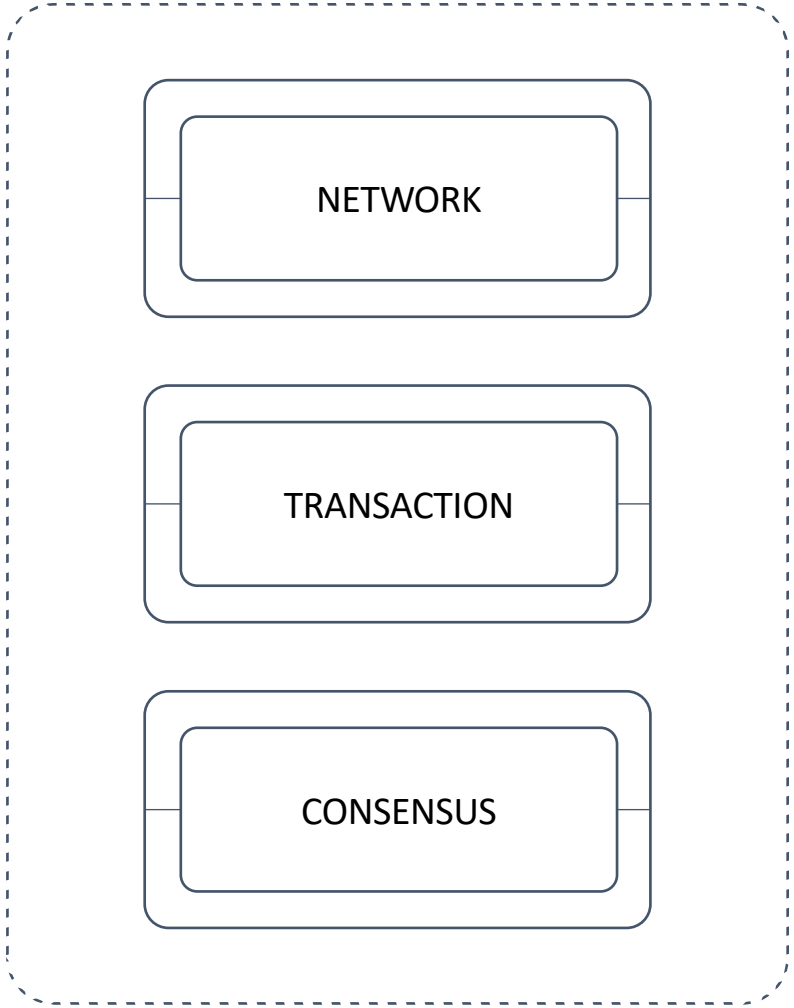
What makes transactions valid

More controversial

Consensus protocol

The way consensus is built around the one chain

The most difficult to change



Blockchain Protocols

Network protocol

Transactions / Blocks download and broadcast

P2P network management (discover, add, remove)

Relatively uncontroversial

Transaction protocol

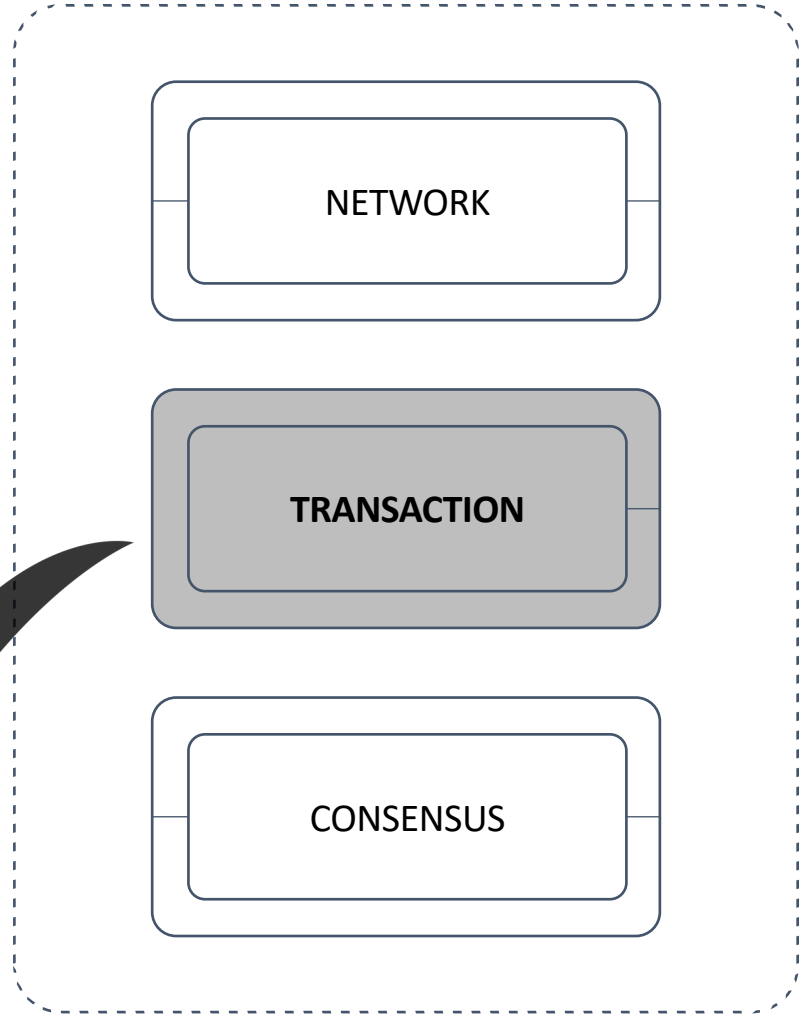
What makes transactions valid

More controversial

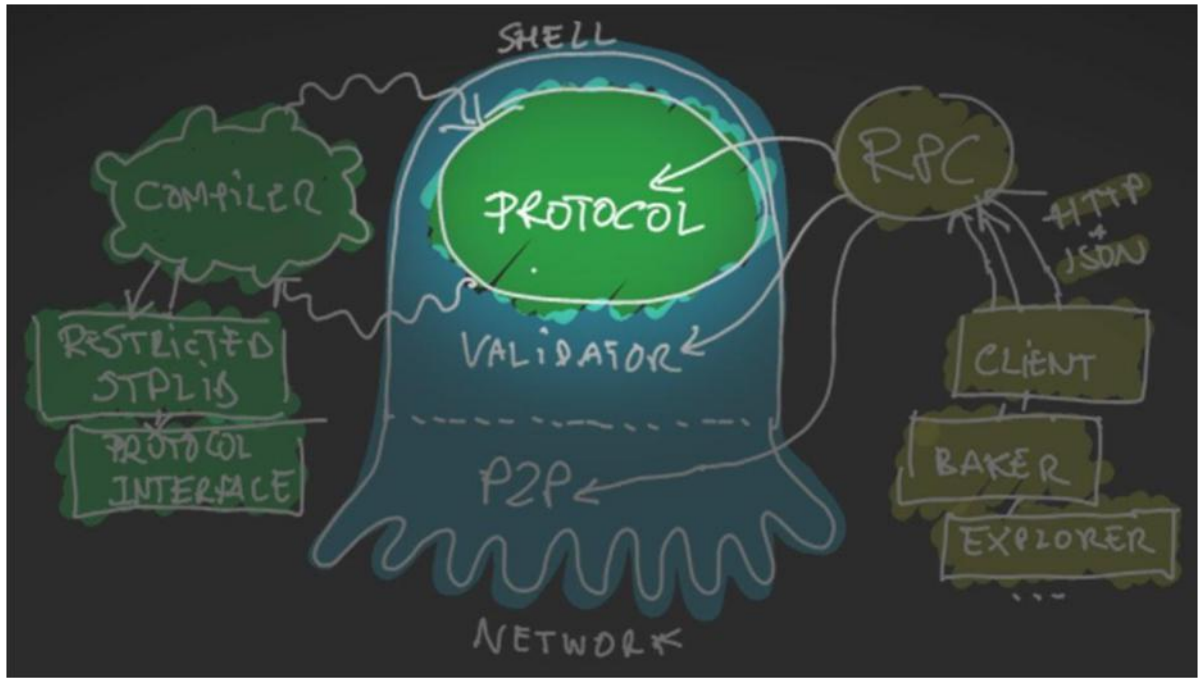
Consensus protocol

The way consensus is built around the one chain

The most difficult to change



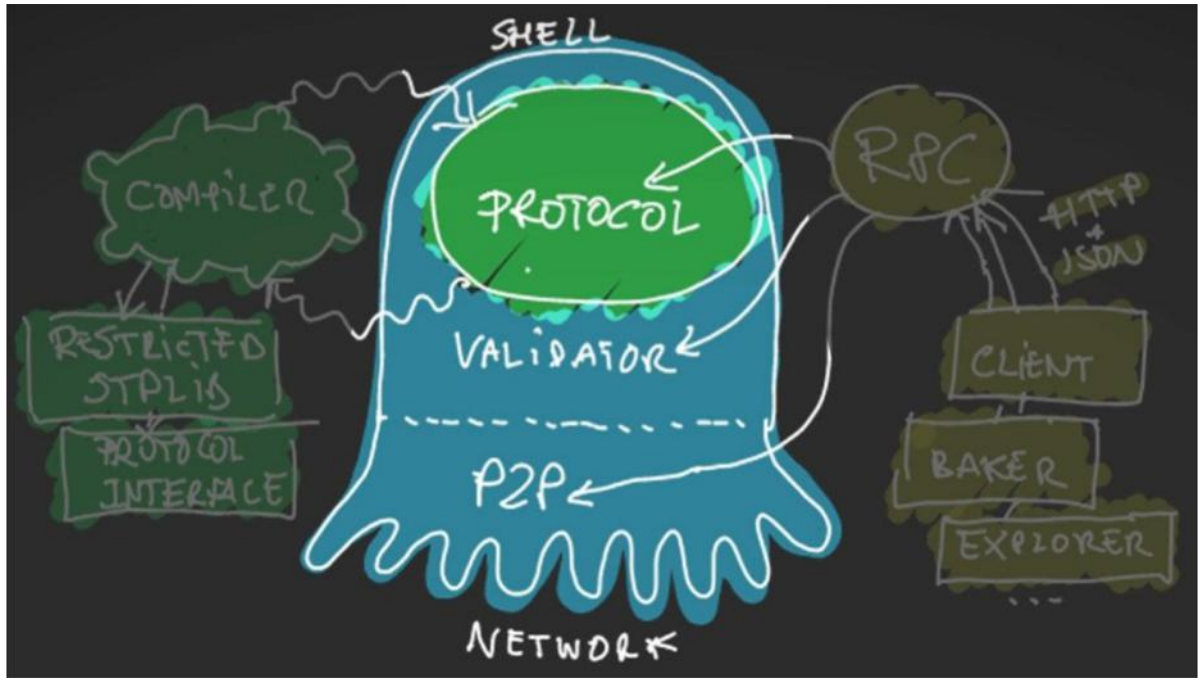
Tezos SW Architecture



Protocol

Tx protocol + consensus protocol
Also called as the economic protocol, block chain protocol, protocol, proto
Self-amendment (plug-ins)
Interpreting operations including tx and blocks
Given only one chain from the shell
Network agnostic

Tezos SW Architecture



Network shell

Running the gossip network + Updating the context

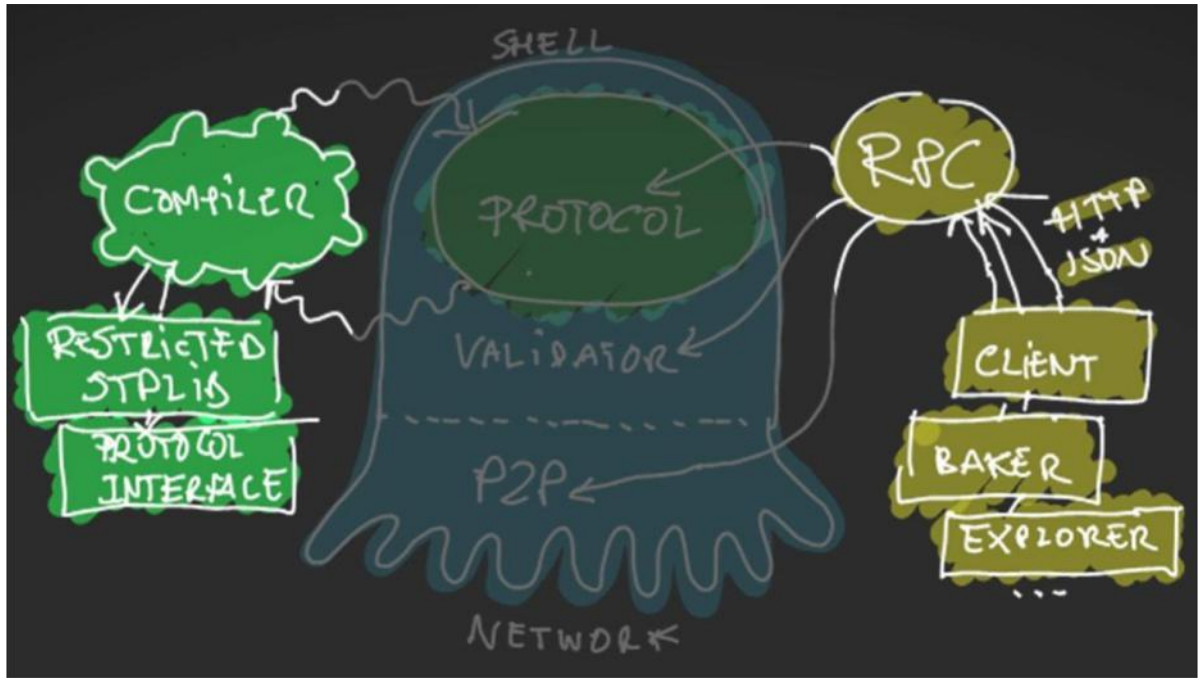
Validator + P2P Layer + Others (the storage of blocks and the state of the ledger)

Maintaining the best chain of highest absolute score (validator)

Managing P2P pool (detects, connects, bans)

Aware of 3 type of objects: transactions(operation), blocks + protocol

Tezos SW Architecture



RPC

The way to interact with the node for the client and third parties

JSON and HTTP

Compiler

Type checking of the protocol's main module

Statically enforced by sandboxing

Getting started with Tezos Node

Main functions

Running the gossip network

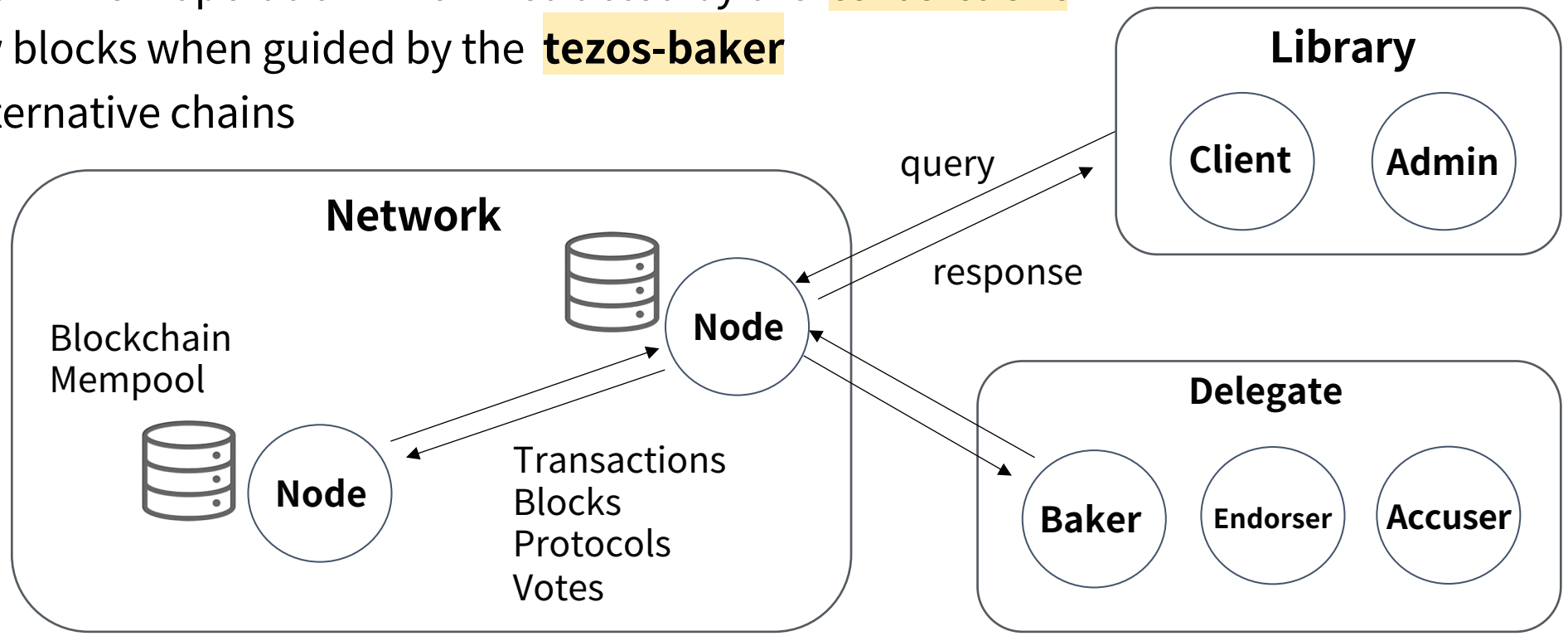
Updating the context

Also

Injecting its own new operation when instructed by the **tezos-client**

Sending new blocks when guided by the **tezos-baker**

Managing alternative chains



Getting started with Tezos Node

Node identity

Let other nodes know itself

Detected by other nodes in the Tezos network.

```
$ ./tezos-node identity generate
```

```
ubuntu@ip-172-31-18-57:~/tezos$ ./tezos-node identity generate
Generating a new identity... (level: 26.00)
Stored the new identity (idsCcz7pFjbgRD6pfixWev1418znik) into '/home/ubuntu/.tezos-node/identity.json'.
ubuntu@ip-172-31-18-57:~/tezos$
```

The identity file in the node's storage.

By default, all block-chain data is stored under **$HOME/.tezos-node/**

```
ubuntu@ip-172-31-18-57:~/tezos$ ls ~/.tezos-node/
identity.json  version.json
```

Getting started with Tezos Node

Running Node

JSON interface is the only interface to the node, but disabled by default.
It can be enabled for the clients to communicate with the node. (**Default port is 8732**)

```
$ ./tezos-node run --rpc-addr  127.0.0.1
```

Syncing

After the node is executed, it starts to find peers and downloads blocks from those peers.

```
ubuntu@ip-172-31-23-117:~/tezos$ ./tezos-node run --rpc-addr 127.0.0.1
Jul  1 04:30:56 - node.main: Starting the Tezos node...
Jul  1 04:30:56 - node.main: No local peer discovery.
Jul  1 04:30:56 - node.main: Peer's global id: idrVn8WZxjWk7Z1TcHTtUN9jwV
UkPz
Jul  1 04:30:56 - main: shell-node initialization: bootstrapping
Jul  1 04:30:56 - main: shell-node initialization: p2p_maintain_started
Jul  1 04:30:56 - validator.block: Worker started
Jul  1 04:30:56 - validation_process.sequential: Initialized
Jul  1 04:30:56 - node.validator: activate chain NetXgtSLGNJvNye
```

Getting started with Tezos **Node**

Node is running

```
ubuntu@ip-172-31-18-57:~/tezos$ ./tezos-node run --rpc-addr 127.0.0.1
Jul 25 10:23:19 - node.main: Starting the Tezos node...
Jul 25 10:23:19 - node.main: No local peer discovery.
Jul 25 10:23:19 - node.main: Peer's global id: idsCcz7pFjbgRD6pfixWev1418znik
Jul 25 10:23:19 - main: shell-node initialization: bootstrapping
Jul 25 10:23:19 - main: shell-node initialization: p2p_maintain_started
Jul 25 10:23:19 - validator.block: Worker started
Jul 25 10:23:19 - validation_process.sequential: Initialized
Jul 25 10:23:19 - node.validator: activate chain NetXgtSLGNJvNye
Jul 25 10:23:19 - validator.chain_1: Worker started for NetXgtSLGNJvN
Jul 25 10:23:19 - p2p.maintenance: Too few connections (0)
Jul 25 10:23:19 - node.chain_validator: no prevalidator filter found for protocol 'Ps6mwMrF2ER2'
Jul 25 10:23:19 - prevalidator.NetXgtSLGNJvN.Ps6mwMrF2ER2_1: Worker started for NetXgtSLGNJvN.Ps6mwMrF2ER2
Jul 25 10:23:19 - node.main: Starting a RPC server listening on ::ffff:127.0.0.1:8732.
Jul 25 10:23:19 - node.main: The Tezos node is now running!
Jul 25 10:23:20 - validator.peer_1: Worker started for NetXgtSLGNJvN:idrm1KhfdmV9
Jul 25 10:23:20 - validator.peer_2: Worker started for NetXgtSLGNJvN:idtdNBn7HYqM
Jul 25 10:23:20 - validator.peer_3: Worker started for NetXgtSLGNJvN:idsohLhiCYYW
Jul 25 10:23:20 - validator.peer_4: Worker started for NetXgtSLGNJvN:idr2piupJd1r
Jul 25 10:23:20 - validator.peer_5: Worker started for NetXgtSLGNJvN:idse7w6uFyRy
Jul 25 10:36:19 - validator.block: Block BMPtRJqFGQJRTfn8bXQR2grLE1M97XnUmG5vgjHMW7St1Wub7Cd successfully validated
Jul 25 10:36:19 - validator.block: Pushed: 2019-07-25T10:36:17-00:00, Treated: 2019-07-25T10:36:17-00:00, Completed:
 2019-07-25T10:36:19-00:00
Jul 25 10:36:19 - node.chain_validator: no prevalidator filter found for protocol 'PsddFKi32cMJ'
Jul 25 10:36:19 - prevalidator.NetXgtSLGNJvN.PsddFKi32cMJ_1: Worker started for NetXgtSLGNJvN.PsddFKi32cMJ
Jul 25 10:36:19 - prevalidator.NetXgtSLGNJvN.Ps6mwMrF2ER2_1: Worker terminated [NetXgtSLGNJvN.Ps6mwMrF2ER2]
Jul 25 10:36:19 - validator.chain_1: Update current head to BMPtRJqFGQJRTfn8bXQR2grLE1M97XnUmG5vgjHMW7St1Wub7Cd (fit
ness 00::0000000000000001), same branch
Jul 25 10:36:19 - validator.chain_1: Pushed: 2019-07-25T10:36:19-00:00, Treated: 2019-07-25T10:36:19-00:00, Complete
d: 2019-07-25T10:36:19-00:00
Jul 25 10:36:25 - validator.block: Block BLwKksYwrxt39exDei7yi47h7aMcVY2kZMZhTwEEoSUwToQUiDV successfully validated
Jul 25 10:36:25 - validator.block: Pushed: 2019-07-25T10:36:25-00:00, Treated: 2019-07-25T10:36:25-00:00, Completed:
 2019-07-25T10:36:25-00:00
Jul 25 10:36:25 - prevalidator.NetXgtSLGNJvN.PsddFKi32cMJ_1: switching to new head BLwKksYwrxt39exDei7yi47h7aMcVY2kZ
MZhTwEEoSUwToQUiDV
```

Getting started with Tezos **Client**

Library

Interacting with the node by **querying** its status or **asking** the node to perform some actions
A built-in wallet

To check the timestamp of the head of the chain (UTC)

```
$ ./tezos-client get timestamp
```

```
ubuntu@ip-172-31-20-138:~$ tezos-client get timestamp
2019-05-31T10:04:27Z
ubuntu@ip-172-31-20-138:~$ tezos-client get timestamp
2019-05-31T10:06:27Z
```

Snapshot mode (ref # 17)

From the genesis to the head

Too long

Few days (SSD) or few weeks (HDD)

Checkpoint

A certain block whose previous blocks are regarded as immutable

Setting a checkpoint to the point where the chain’s security is not compromised

Storing the data after the checkpoint dramatically cut the time of syncing

Make sure tezos-node is not running before run this command.

```
$ curl "https://gitlab.com/tezoskorea/tezos-snapshot/raw/master/quicksync.sh" | bash -s alphanet
```

Getting started with Tezos **Client**

Generate an Address

tezos-client has a **built-in wallet**

To generate a new address (a pair of keys)

```
$ ./tezos-client gen keys tezoskorea –s ed25519 --encrypted
```

```
ubuntu@ip-172-31-18-57:~/tezos$ ./tezos-client gen keys myWallet --encrypted --force
Enter password to encrypt your key:
Confirm password:
```

Tezos supports 3 DSA schemes

ed25519 curve: tz1, recommended, there is a **verified library** (performance, security)

secp256k1 curve: tz2, the one used in Bitcoin and Ethereum, **HSM support**

secp256k1 curve: tz3, P-256, **HSM support**

Getting started with Tezos **Client**

Generate an Address

tezos-client has a **built-in wallet**

To generate a new address (a pair of keys)

```
$ ./tezos-client gen keys tezoskorea –s ed25519 --encrypted
```

```
ubuntu@ip-172-31-18-57:~/tezos$ ./tezos-client gen keys myWallet --encrypted --force
Enter password to encrypt your key:
Confirm password:
```

Never forget your password.

tezoskorea is an alias for your addresses only used locally.

You can freely set an alias and use this alias to check the balance and transfer tokens.

```
$ ./tezos-client get balance for myWallet
```

```
ubuntu@ip-172-31-18-57:~/tezos$ tezos-client get balance for myWallet
0 ꜩ
```

Getting started with Tezos **Client**

Key management

$ ./tezos-client list known addresses

```
ubuntu@ip-172-31-28-211:~$ tezos-client list known addresses
john: tz1LsDttshWsAAPHZqJGYuSZfrkGMcSpZ25B (encrypted sk known)
kim: tz1M7sFiTqDffL5Nj4znQcXixcVwZEbW7Y1z (unencrypted sk known)
```

$ ./tezos-client show address **tezoskorea** --show-secret

```
ubuntu@ip-172-31-18-57:~/tezos$ tezos-client show address myWallet --show-secret
Hash: tz1PJzmWptPmpoNnTZ2exsTd7aEQfhtoGdvU
Public Key: edpkub5rGi59z3HxKqEbuUT7xk6PxCVGUDcEokPCLd9emQ5aw6BZuQ
Secret Key: encrypted:edesk1QKMCeyvQUYJSJzGf4RPeqCj9KHiQGgad4CzSr2eB4pcQvnKpcZwJv6TEGSyVUcPoJRD35vQE3H5Tepjp8r
```

The directory directory **~/.tezos-client** is filled with 3 files

public_key_hashs , public_keys and secret_keys .

```
ubuntu@ip-172-31-18-57:~/tezos$ ls ~/.tezos-client/
public_key_hashs  public_keys  secret_keys
```

```
ubuntu@ip-172-31-18-57:~/tezos$ cat ~/.tezos-client/secret_keys && echo
[ { "name": "myWallet",
    "value":
      "encrypted:edesk1QKMCeyvQUYJSJzGf4RPeqCj9KHiQGgad4CzSr2eB4pcQvnKpcZwJv6TEGSyVUcPoJRD35vQE3H5Tepjp8r" } ]
```

Free tez from faucet (ref #18)

faucet.tzalpha.net
alphanet only

Free tez from faucet (ref #18)

faucet.tzalpha.net

alphanet only

$ ./tezos-client activate account **faucet** with tz1__xxxxxx__.json

```
ubuntu@ip-172-31-18-57:~/tezos$ ./tezos-client activate account faucet with tz1duCT9DkNUiYnDY1yKmP
Txp6C15QFBW5fx.json
Node is bootstrapped, ready for injecting operations.
Operation successfully injected in the node.
Operation hash is 'ooRvmCbi7ztqU3bxX5G2185s5puXe2dkQ9os8qRkzEBdrAvUvkf'
Waiting for the operation to be included...
Operation found in block: BMC8ufZvFfcxtFt6K2WUjfWhhsUUdy7HFroCWqiepQLWhyNVtxR (pass: 2, offset: 0)
This sequence of operations was run:
  Genesis account activation:
    Account: tz1duCT9DkNUiYnDY1yKmPTxp6C15QFBW5fx
    Balance updates:
      tz1duCT9DkNUiYnDY1yKmPTxp6C15QFBW5fx ... +ꜩ10113.895767

The operation has only been included 0 blocks ago.
We recommend to wait more.
Use command
  tezos-client wait for ooRvmCbi7ztqU3bxX5G2185s5puXe2dkQ9os8qRkzEBdrAvUvkf to be included --confi
rmations 30 --branch BMT1wim2ef7cyXE8ZJ7XSdVDgb9dzK2h18J7QcBuwK5E9sn96XN
and/or an external block explorer.
Account faucet (tz1duCT9DkNUiYnDY1yKmPTxp6C15QFBW5fx) activated with ꜩ10113.895767.
```

Free tez from faucet (ref #18)

faucet.tzalpha.net

alphanet only

$ ./tezos-client list known addresses

```
ubuntu@ip-172-31-18-57:~/tezos$ ./tezos-client list known addresses
faucet: tz1duCT9DkNUiYnDY1yKmPTxp6C15QFBW5fx (unencrypted sk known)
myWallet: tz1PJzmWptPmpoNnTZ2exsTd7aEQfhtoGdvU (encrypted sk known)
```

$ ./tezos-client get balance for **faucet**

```
ubuntu@ip-172-31-18-57:~/tezos$ ./tezos-client get balance for faucet
10113.895767 ꜩ
```

Injecting a transaction

Transactions

transfer command returns a receipt with all the information of the transaction

tezos-client then waits for the operation to be included in one block

30 blocks or more is recommended

tezos-node can validate an operation before injecting it to the network (**--dry-run**)

Gas and storage limits can be determined by this validation

```
$ ./tezos-client transfer 1 from tezoskorea to john --force-low-fee -D
$ ./tezos-client wait for OPHASH to be included --confirmations 30
```

Injecting a transaction

Transactions **at first**

$ ./tezos-client transfer 1 from **myWallet** to john --force-low-fee --dry-run

```
ubuntu@ip-172-31-28-211:~$ tezos-client transfer 1 from myWallet to john
--force-low-fee --dry-run
Node is bootstrapped, ready for injecting operations.
Fatal error:
  The operation will burn ꜩ0.257 which is higher than the configured burn
 cap (ꜩ0).
   Use `--burn-cap 0.257` to emit this operation.
```

Nobody in the network knows **the public key** of **tezoskorea**

A **reveal** operation is an operation that writes on the chain the public key associated with a public key hash for an implicit account.

Injecting a transaction

Transactions **receipt**

```
ubuntu@ip-172-31-18-57:~/tezos$ ./tezos-client transfer 10112 from faucet to myWallet --burn-cap 0.257 --force-low-fee
Node is bootstrapped, ready for injecting operations.
Estimated gas: 10200 units (will add 100 for safety)
Estimated storage: 257 bytes added (will add 20 for safety)
Operation successfully injected in the node.
Operation hash is 'ooHS29Bu1hazmbqYRcfKT4HjxD5ywcwow8JY7q5ugWMGXqoLBrb'
Waiting for the operation to be included...
Operation found in block: BLzrTp8UjPNsEc6coig4W2YLamKWrqEsbc6imKWaKiGWWW4gTg5 (pass: 3, offset: 6)
This sequence of operations was run:
  Manager signed operations:
    From: tz1duCT9DkNUiYnDY1yKmPTxp6C15QFBW5fx
    Fee to the baker: ꜩ0.00126
    Expected counter: 176192
    Gas limit: 10000
    Storage limit: 0 bytes
    Balance updates:
      tz1duCT9DkNUiYnDY1yKmPTxp6C15QFBW5fx ..... -ꜩ0.00126
      fees(tz3gN8NTLNLJg5KRsUU47NHNVHbdhcFXjjaB,256) ... +ꜩ0.00126
    Revelation of manager public key:
      Contract: tz1duCT9DkNUiYnDY1yKmPTxp6C15QFBW5fx
      Key: edpkuv35rRDMrnMnSHPauLTe61XMsRBXe4b4xrr4opjgTiBppzB6Qk
```

RPC API (ref #19)

Usage

./tezos-node run --rpc-addr <ADDR:PORT>

(Default port is 8732)

./tezos-client rpc get list

./tezos-client rpc get <URL>

./tezos-client rpc post <URL> with <JSON>

```
ubuntu@ip-172-31-23-117:~/tezos$ ./tezos-client rpc list

Available services:
 + chains/<chain_id>/
   - GET /chains/<chain_id>/blocks
       Lists known heads of the blockchain sorted with decreasing fitness.
       Optional arguments allows to returns the list of predecessors for
       known heads or the list of predecessors for a given list of blocks.
   - /chains/<chain_id>/blocks/<block_id> <dynamic>
   - GET /chains/<chain_id>/chain_id
       The chain unique identifier.
   - GET /chains/<chain_id>/checkpoint
       The current checkpoint for this chain.
   - GET /chains/<chain_id>/invalid_blocks
       Lists blocks that have been declared invalid along with the errors
       that led to them being declared invalid.
   - GET /chains/<chain_id>/invalid_blocks/<block_hash>
       The errors that appears during the block (in)validation.
   - DELETE /chains/<chain_id>/invalid_blocks/<block_hash>
       Remove an invalid block for the tezos storage
   - /chains/<chain_id>/mempool <dynamic>
 - GET /errors
     Schema for all the RPC errors from the shell
 - GET /fetch_protocol/<Protocol_hash>
     Fetch a protocol from the network.
 + injection/
   - POST /injection/block
       Inject a block in the node and broadcast it. The `operations`
       embedded in `blockHeader` might be pre-validated using a contextual
       RPCs from the latest block (e.g. '/blocks/head/context/preapply').
       Returns the ID of the block. By default, the RPC will wait for the
       block to be validated before answering.
   - POST /injection/operation
       Inject an operation in node and broadcast it. Returns the ID of the
       operation. The `signedOperationContents` should be constructed using
       a contextual RPCs from the latest block and signed by the client. By
       default, the RPC will wait for the operation to be (pre-)validated
       before answering. See RPCs under /blocks/prevalidation for more
       details on the prevalidation context.
```

RPC API (ref #19)

Retrieve constants

```
$ ./tezos-client rpc get /chains/main/blocks/head/context/constants | jq
```

```
ubuntu@ip-172-31-23-117:~/tezos$ ./tezos-client rpc get /chains/main/blocks/head/context/constants | jq
{
  "proof_of_work_nonce_size": 8,
  "nonce_length": 32,
  "max_revelations_per_block": 32,
  "max_operation_data_length": 16384,
  "max_proposals_per_delegate": 20,
  "preserved_cycles": 3,
  "blocks_per_cycle": 2048,
  "blocks_per_commitment": 32,
  "blocks_per_roll_snapshot": 256,
  "blocks_per_voting_period": 8192,
  "time_between_blocks": [
    "30",
    "40"
  ],
  "endorsers_per_block": 32,
  "hard_gas_limit_per_operation": "400000",
  "hard_gas_limit_per_block": "4000000",
  "proof_of_work_threshold": "70368744177663",
  "tokens_per_roll": "10000000000",
  "michelson_maximum_type_size": 1000,
  "seed_nonce_revelation_tip": "125000",
  "origination_size": 257,
  "block_security_deposit": "512000000",
  "endorsement_security_deposit": "64000000",
  "block_reward": "16000000",
  "endorsement_reward": "2000000",
  "cost_per_byte": "1000",
  "hard_storage_limit_per_operation": "60000"
}
```

RPC API (ref #19)

Retrieve a block

```
$ ./tezos-client rpc get /chains/main/blocks/<head | hash | level>
```

```
ubuntu@ip-172-31-23-117:~/tezos$ ./tezos-client rpc get /chains/main/blocks/head | jq
{
  "protocol": "PsddFKi32cMJ2qPjf43Qv5GDWLDPZb3T3bF6fLKiF5HtvHNU7aP",
  "chain_id": "NetXgtSLGNJvNye",
  "hash": "BLW8dwpYrVmgT2yX1n9R5PsJ2qEzpEQsvFoKbCDFpKiixX2sywh",
  "header": {
    "level": 38208,
    "proto": 1,
    "predecessor": "BL2o2N7W2K7VH8zn2XXCn4aaNXk2dMqZvvH8xTRdTK5P7skrNUo",
    "timestamp": "2018-12-17T16:11:48Z",
    "validation_pass": 4,
    "operations_hash": "LLoZtoSQur4M7dWAVHWYvNXzvcRxMkTi1qh91EmspVK3chJ18j6KQ",
    "fitness": [
      "00",
      "000000000010f093"
    ],
    "context": "CoW3CLeMriNtEvon6vBUv993RKbYKN1WStPd7XaumWVamujCrTiS",
    "priority": 0,
    "proof_of_work_nonce": "000000038ceee095",
    "seed_nonce_hash": "nceUnn2PpKfwE8RTw1siMWWbhW7h42Qv3ymK4npQXjcD857KXn4o2",
    "signature": "sigaiWaMAciJPFh2mSFCHbbsz6K9cxHi8EqAbHSy3favYPv1WaVKx97skTFgFHDDZV8mdst
1ntPYAoJU15dgTgpq4K6CkU5y"
  },
```

RPC API **before the checkpoint** (ref #19)

Public Node1 - tezTech

```
$ curl https://mainnet.tezrpc.me/chains/main/blocks/20 | jq
```

```
choemincheols-MacBook-Pro:tezos mcwithimp$ curl https://mainnet.tezrpc.me/chains/main/blocks/541889/operations/1  | jq
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   509    0   509    0     0    884      0 --:--:-- --:--:-- --:--:--   883
[
  {
    "protocol": "Pt24m4xiPbLDhVgVfABUjirbmda3yohdN82Sp9FeuAXJ4eV9otd",
    "chain_id": "NetXdQprcVkpaWU",
    "hash": "ooPk8beQUTNztpSb46hH3ta6wJw8Akbx99GhpMDfMEfgpCf9QoF",
    "branch": "BLR6qfNRMQeux8tdrYWu674zKs7Kheyd5zb1rSBWFzZGty8x89B",
    "contents": [
      {
        "kind": "proposals",
        "source": "tz1PPUo28B8BroqmVCMMNDudG4ShA2bzicrU",
        "period": 16,
        "proposals": [
          "PsBABY5nk4JhdEv1N1pZbt6m6ccB9BfNqa23iKZcHBh23jmRS9f"
        ],
        "metadata": {}
      }
    ],
    "signature": "sigu9LLGFPb23svgC1VaVKFMEPAQ1Nk4RBuDZjQXZt1qLVteF6qqTJHscZktW8PTruzmHzTkTjFdLg6FV8HTLq7Q1rW3FHNi"
  }
]
```

RPC API **before the checkpoint** (ref #19)

Public Node2 - tzscan

```
$ ./tezos-client -A alphanet-node.tzscan.io -P 443 -S rpc get /chains/main/blocks/20
```

```
choemincheols-MacBook-Pro:tezos mcwithimp$ tezos-client -A mainnet-node.tzscan.io -P 443 -S rpc get /chains/main/
blocks/541889/operations/1
[ { "protocol": "Pt24m4xiPbLDhVgVfABUjirbmda3yohdN82Sp9FeuAXJ4eV9otd",
    "chain_id": "NetXdQprcVkpaWU",
    "hash": "ooPk8beQUTNztpSb46hH3ta6wJw8Akbx99GhpMDfMEfgpCf9QoF",
    "branch": "BLR6qfNRMQeux8tdrYWu674zKs7Kheyd5zb1rSBWFzZGty8x89B",
    "contents":
      [ { "kind": "proposals",
          "source": "tz1PPUo28B8BroqmVCMMNDudG4ShA2bzicrU", "period": 16,
          "proposals":
            [ "PsBABY5nk4JhdEv1N1pZbt6m6ccB9BfNqa23iKZcHBh23jmRS9f" ],
          "metadata": {} } ],
    "signature":
      "sigu9LLGFPb23svgC1VaVKFMEPAQ1Nk4RBuDZjQXZt1qLVteF6qqTJHscZktW8PTruzmHzTkTjFdLg6FV8HTLq7Q1rW3FHNi" } ]
```

Block structure

A Block

consists of ***protocol***, ***chain_id***, ***hash***, ***header***, ***metadata*** and ***operations***.


```
{
   "protocol":"Pt24m4xiPbLDhVgVfABUjirbmda3yohdN82Sp9FeuAXJ4eV9otd",
   "chain_id":"NetXdQprcVkpaWU",
   "hash":"BM45PLkPkvSmh7L5XgyrRBYjHvjYWk1icpqoh2S5aYuoyAkz5AN",
   "header":{…},
   "metadata":{…},
   "operations":[…]
}
```


The ***header*** contains **height, previous hash, merkle root** and etc.


```
"header":{
   "level":503443,
   "proto":4,
   "predecessor":"BKpumgUMxfMTwf8kRxNKaBChGWU4FcZv188cnDZmHEqkzJHHZyW",
   "timestamp":"2019-07-01T12:58:28Z",
   "validation_pass":4,
   "operations_hash":"LLoZVdpcmXK6w8wfhJnadHWKvhhCfrjFeWSQRs9p1FhJUHbTU2mgG",
   "fitness":[
      "00",
      "0000000000f16761"
   ],
   "context":"CoWD4rgFcjnZzvL1ZDmT8Z7a9JvstSJMyTriHMGzuMddPaChC51y",
   "priority":0,
   "proof_of_work_nonce":"000000035756c153",
   "signature":"sigvJLV275L1hH1Rup49URnYj464TeeTnGunirbcVkijriyvfgNNqcye2pJYvXBfjQ7cwMBbUZYroXvzbGfPGyzXNLZ6kTtm"
},
```

Block structure

A Block

consists of ***protocol***, ***chain_id***, ***hash***, ***header***, ***metadata*** and ***operations***.


```
{
    "protocol":"Pt24m4xiPbLDhVgVfABUjirbmda3yohdN82Sp9FeuAXJ4eV9otd",
    "chain_id":"NetXdQprcVkpaWU",
    "hash":"BM45PLkPkvSmh7L5XgyrRBYjHvjYWk1icpqoh2S5aYuoyAkz5AN",
    "header":{…},
    "metadata":{…},
    "operations":[…]
}
```


The ***metadata*** contains **baker, baking reward** and etc.


```
"metadata":{
   "protocol":"Pt24m4xiPbLDhVgVfABUjirbmda3yohdN82Sp9Feu
   "next_protocol":"Pt24m4xiPbLDhVgVfABUjirbmda3yohdN82S
   "test_chain_status":{…},
   "max_operations_ttl":60,
   "max_operation_data_length":16384,
   "max_block_header_length":238,
   "max_operation_list_length":[…],
   "baker":"tz1Yju7jmmsaUiG9qQLoYv35v5pHgnWoLWbt",
   "level":{…},
   "voting_period_kind":"testing_vote",
   "nonce_hash":null,
   "consumed_gas":"33225",
   "deactivated":[…],
   "balance_updates":[…]
},
```



```
"balance_updates":[
   {
      "kind":"contract",
      "contract":"tz1Yju7jmmsaUiG9qQLoYv35v5pHgnWoLWbt",
      "change":"-512000000"
   },
   {
      "kind":"freezer",
      "category":"deposits",
      "delegate":"tz1Yju7jmmsaUiG9qQLoYv35v5pHgnWoLWbt",
      "cycle":122,
      "change":"512000000"
   },
   {
      "kind":"freezer",
      "category":"rewards",
      "delegate":"tz1Yju7jmmsaUiG9qQLoYv35v5pHgnWoLWbt",
      "cycle":122,
      "change":"16000000"
   }
]
```

Block structure

Operations

Operators changing the global state.

1st list: the **endorsements**.

2nd list: the operations regarding **votes and proposals**.

3rd list: anonymous operations.

The last list: the **manager operations** (reveal, transaction, delegation and origination).

Endorsement

Block structure

Manager operations

A manager operation has 3 important parameters: counter, gas limit and storage limit.

Counter

Each **account** has the counter.
Making each operation **unique** and **applied only once in order**.

Gas / Storage limit

An operation cannot consume more than these limits

Transactions

Messages between accounts
Signed by the private key of a manager with counter
Trigger for a smart contract to work