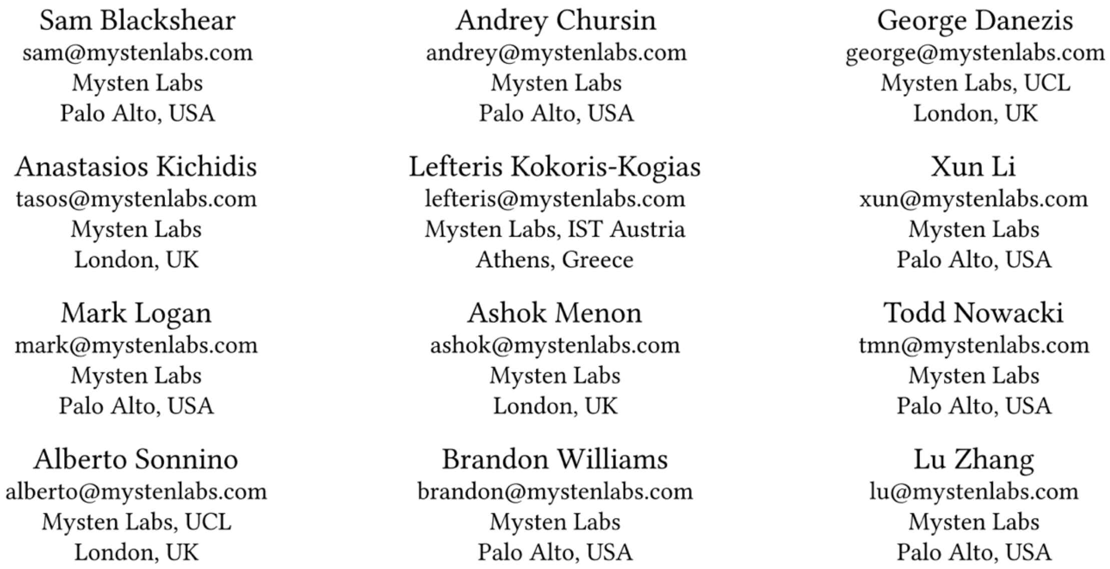
Lutris: A Blockchain Combining Broadcast and Consensus

CCS 24



Sam Blackshear
sam@mystenlabs.com
Mysten Labs
Palo Alto, USA

Andrey Chursin
andrey@mystenlabs.com
Mysten Labs
Palo Alto, USA

George Danezis
george@mystenlabs.com
Mysten Labs, UCL
London, UK

Anastasios Kichidis
tasos@mystenlabs.com
Mysten Labs
London, UK

Lefteris Kokoris-Kogias
lefteris@mystenlabs.com
Mysten Labs, IST Austria
Athens, Greece

Xun Li
xun@mystenlabs.com
Mysten Labs
Palo Alto, USA

Mark Logan
mark@mystenlabs.com
Mysten Labs
Palo Alto, USA

Ashok Menon
ashok@mystenlabs.com
Mysten Labs
London, UK

Todd Nowacki
tmn@mystenlabs.com
Mysten Labs
Palo Alto, USA

Alberto Sonnino
alberto@mystenlabs.com
Mysten Labs, UCL
London, UK

Brandon Williams
brandon@mystenlabs.com
Mysten Labs
Palo Alto, USA

Lu Zhang
lu@mystenlabs.com
Mysten Labs
Palo Alto, USA

2024. 11. 25

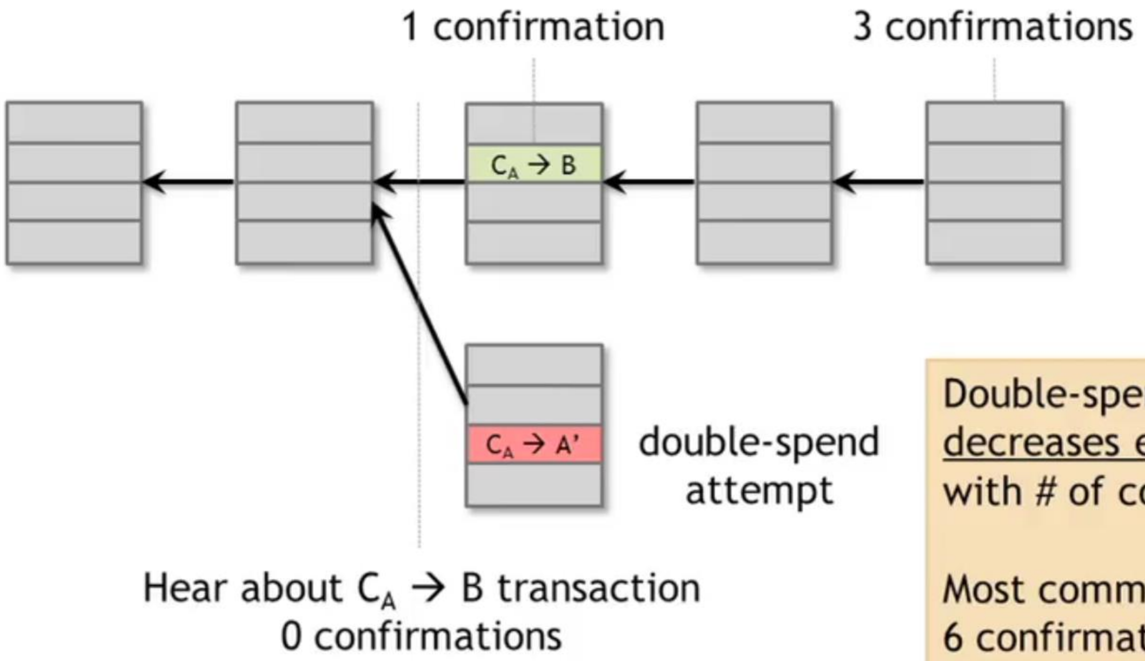
서울대 분산시스템연구실

석사과정 문보설

Mysten Labs

- Zef: Low-latency, Scalable, Private Payments
- Twins: BFT Systems Made Robust (OPODIS ‘21)
- SybilQuorum: Open Distributed Ledgers Through Trust Networks
- **Narwhal and Tusk: A DAG-based Mempool and Efficient BFT Consensus (EuroSys ‘22)**
- HammerHead: Score-based Dynamic Leader Selection (ICDCS ‘24)
- **FastPay: High-Performance Byzantine Fault Tolerant Settlement (AFT ‘20)**
- **Bullshark: DAG BFT Protocols Made Practical**
- Be Aware of Your Leaders (FC ‘22)
- zkLogin: Privacy-Preserving Blockchain Authentication with Existing Credentials
- **Sui Lutris: A Blockchain Combining Broadcast and Consensus (CCS ‘24)**
- **Mysticeti: Reaching the Limits of Latency with Uncertified DAGs (NDSS ‘25)**
- **Mahi-Mahi: Low-Latency Asynchronous BFT DAG-Based Consensus**

Double Spending and Consensus

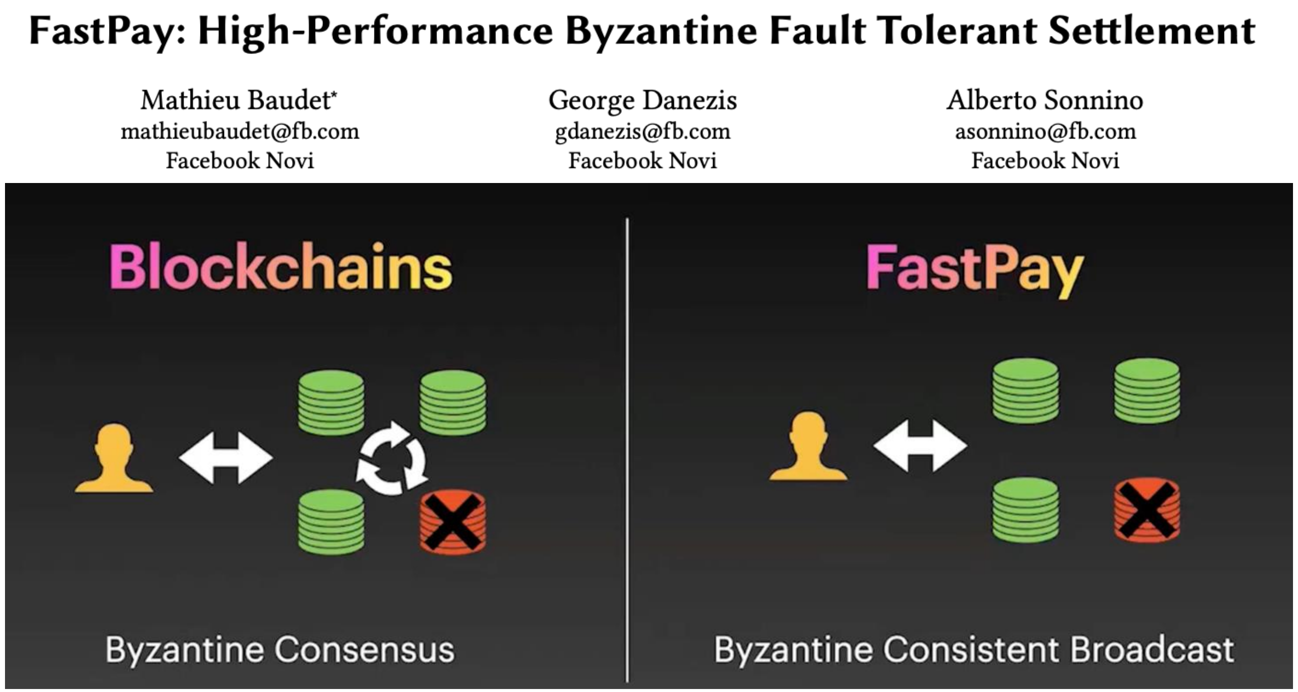


Double-spend probability decreases exponentially with # of confirmations

Most common heuristic:
6 confirmations

Consensus-less Blockchain

- Consensus-less blockchain (Zef(WPES '23), **FastPay(AFT '20)**, Astro(DSN '20)…)
  - Utilize **consistent broadcast** to forgo consensus



FastPay: High-Performance Byzantine Fault Tolerant Settlement

Mathieu Baudet*
mathieubaudet@fb.com
Facebook Novi

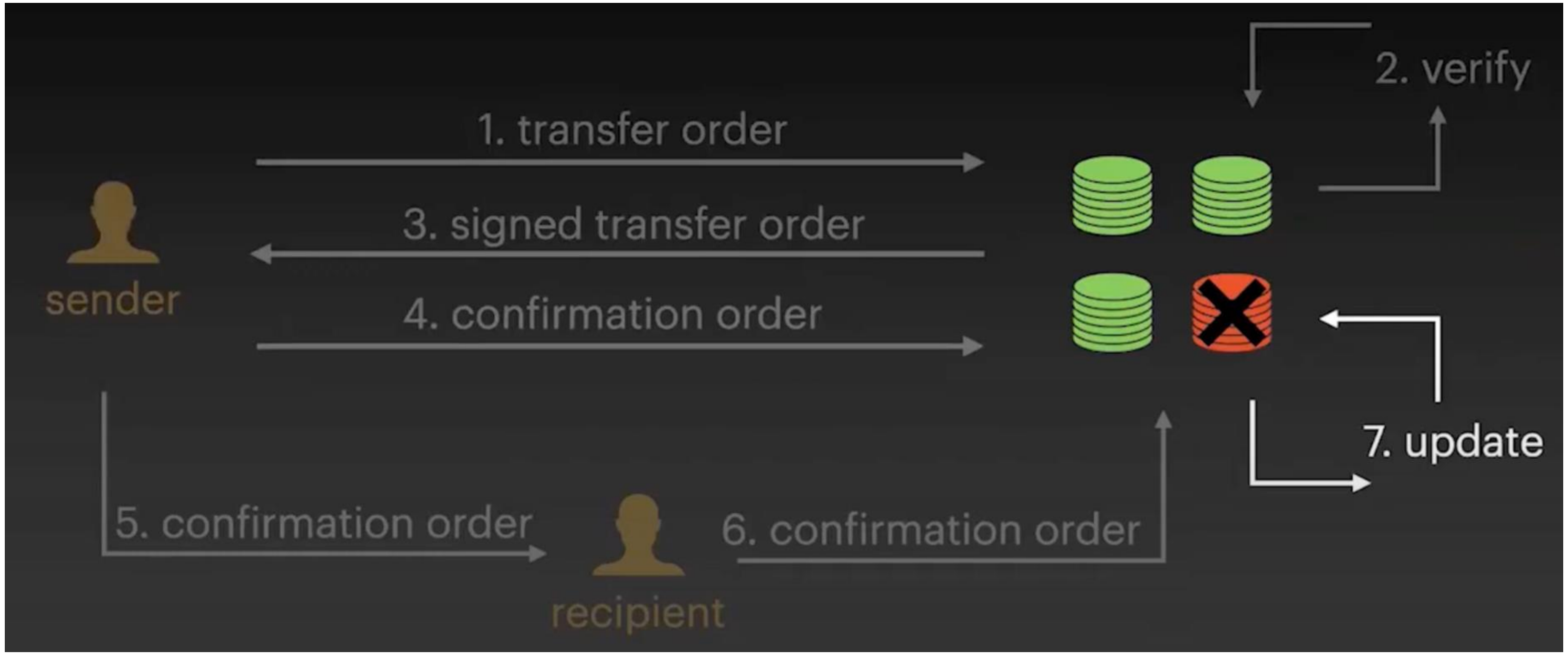
George Danezis
gdanezis@fb.com
Facebook Novi

Alberto Sonnino
asonnino@fb.com
Facebook Novi

FastPay

- UTXO Model
  - Single writer model
  - Owner만 수정 가능
  - Move의 Owned Object

- Key Concept
  - Consensus 대신 Owner가 Tx ordering
  - Tx 실행 조건: 2f + 1 validator signature 수집 (cert)
  - 더블스펜딩 시도 -> 자산 잠금



Problems in Consensus-less Blockchain

1. Limited to asset transfers
   - Account model에서의 자산은 대부분 shared object
   - Shared object: 여러 당사자가 접근 및 소유하는 대상
     - Smart Contract
     - Multi-owned object
     - Multi-writer
   - Shared object의 경우 실행 순서에 따라 결과값이 달라질 수 있음
   - 당사자끼리의 Sequence Coordination이 어려움
     - 대부분의 블록체인이 이 문제를 해결하기 위해 consensus라는 제 3의 sequencing mechanism을 두는 것
2. Do not support state checkpoints
3. Equivocations lock the assets forever

Problems in Consensus-less Blockchain

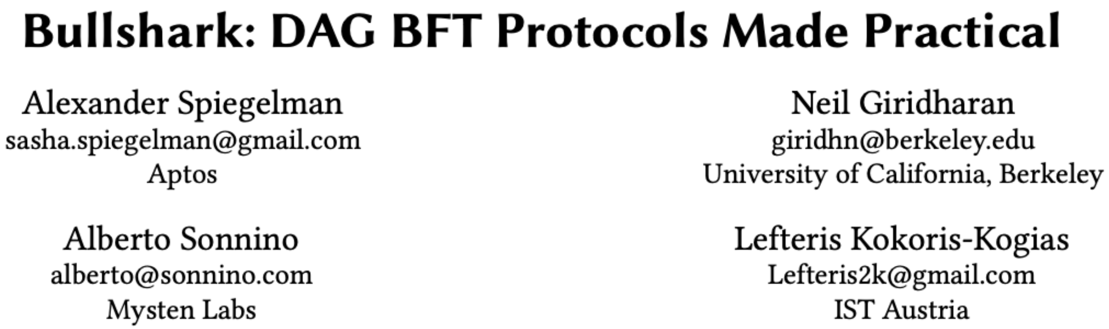
1. Limited to asset transfers
2. Do not support state checkpoints
    - 새로운 validator의 bootstrapping(sync)에 필요
    - Block header, transaction hashes …
3. Equivocations lock the assets forever

Problems in Consensus-less Blockchain

1. Limited to asset transfers
2. Do not support state checkpoints
3. Equivocations lock the assets forever (Freeze)
    - 유저가 상충하는 요청을 보낼 경우 자산이 영원히 잠김
    - Lock이 걸려있는 자산에 write를 요청하는 경우
    - ex) Coin A를 사용하기로 약속(Lock)한 시점에 coin A를 사용하는 또 다른 요청을 보내는 경우

Consensus-based Blockchain

- Latency & complexity
- Redundancy
  - Consensus가 필요하지 않은 tx조차 consensus를 거치게 설계
  - Consensus가 필요하지 않은 tx == Owned Obeject에 대한 tx
    - Fastpay처럼 User가 직접 ordering 가능



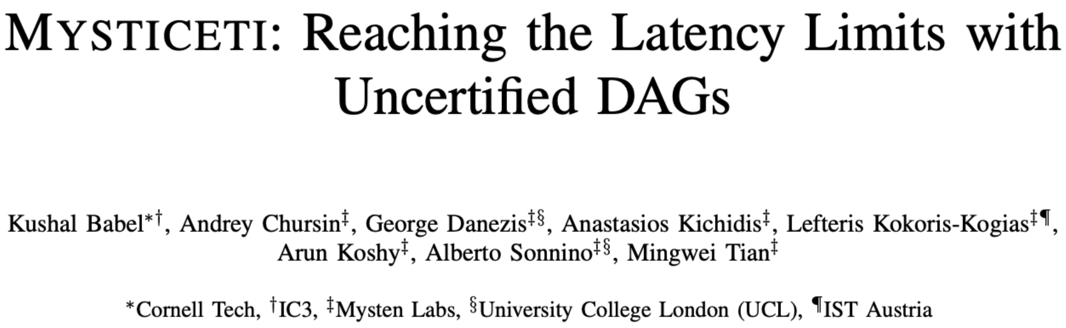
Bullshark: DAG BFT Protocols Made Practical

Alexander Spiegelman
sasha.spiegelman@gmail.com
Aptos

Neil Giridharan
giridhn@berkeley.edu
University of California, Berkeley

Alberto Sonnino
alberto@sonnino.com
Mysten Labs

Lefteris Kokoris-Kogias
Lefteris2k@gmail.com
IST Austria

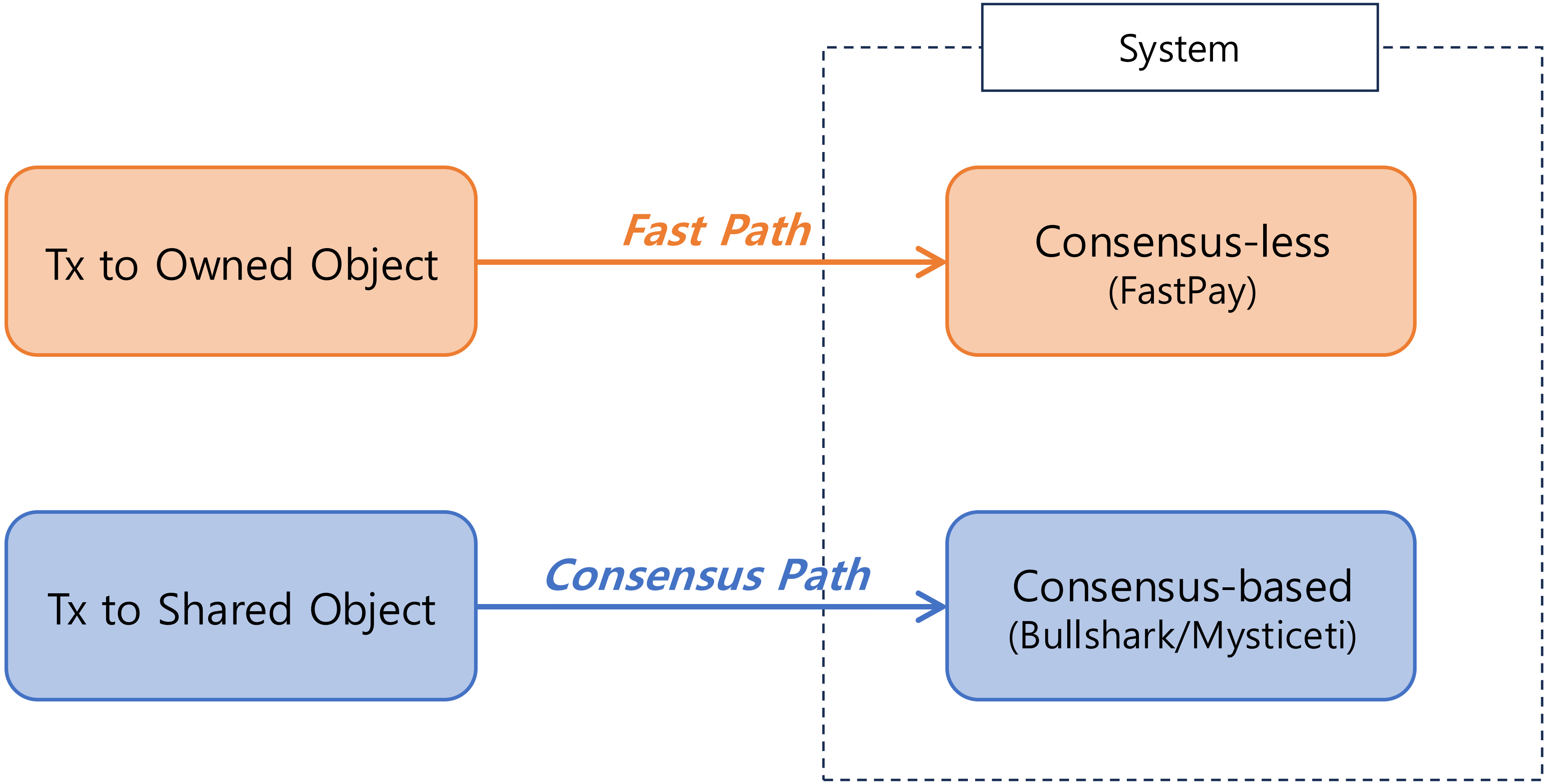


MYSTICETI: Reaching the Latency Limits with Uncertified DAGs

Kushal Babel[*†], Andrey Chursin[‡], George Danezis[‡§], Anastasios Kichidis[‡], Lefteris Kokoris-Kogias[‡¶], Arun Koshy[‡], Alberto Sonnino[‡§], Mingwei Tian[‡]

[*]Cornell Tech, [†]IC3, [‡]Mysten Labs, [§]University College London (UCL), [¶]IST Austria

Combining Fast Path + Consensus Path

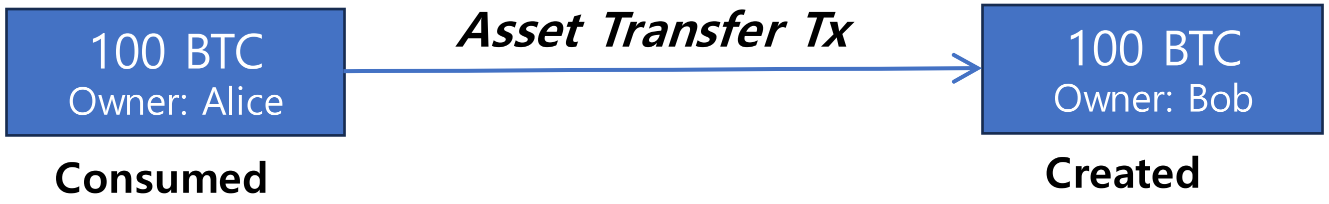


Contribution

1. First smart contract system combining consensus-less mode and consensus-based mode
2. Support checkpointing in consensus-less blockchain
3. Forgive equivocation during reconfiguration process
4. Provide a production-grade evaluation of the system

Move language

- UTXO 모델
  - Owned Object



100 BTC
Owner: Alice

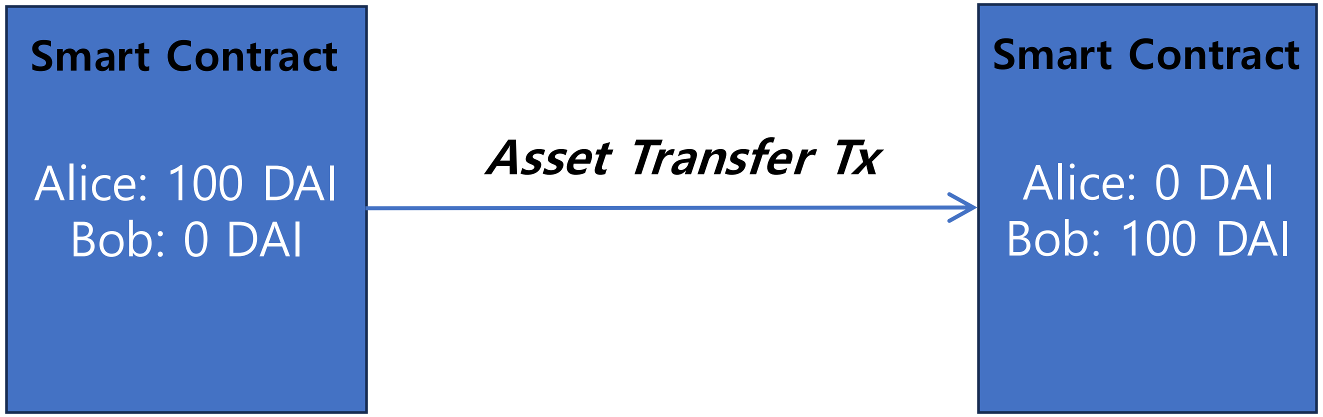
Consumed

Asset Transfer Tx

100 BTC
Owner: Bob

Created

- Account-based 모델
  - Shared Object



Smart Contract

Alice: 100 DAI
Bob: 0 DAI

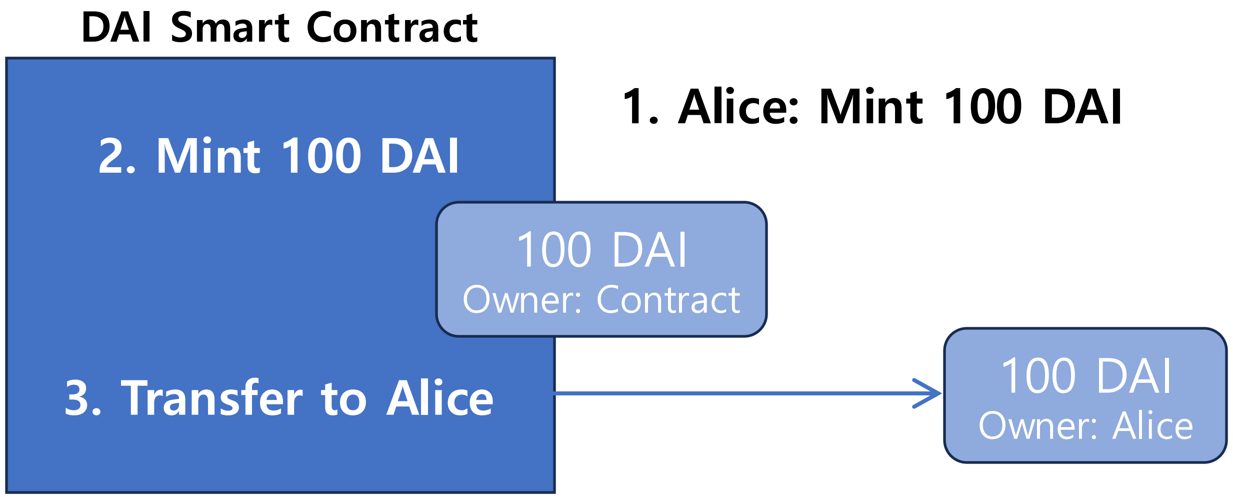
Asset Transfer Tx

Smart Contract

Alice: 0 DAI
Bob: 100 DAI

Move language

- **Object Model**
  - **Owned Object (Assets)**
  - **Shared Object (Smart contracts)**
- Combine UTXO model & Account based model



- Version ID
- Object ID
- Object Key: Version ID + Object ID

Move language

Parallelizable

Irrelevant to the order

Transfer 100 DAI Owner: Alice **to Bob**

Transfer 20 DAI Owner: Carol **to Bob**

Need Consensus

Result will be affected by the order

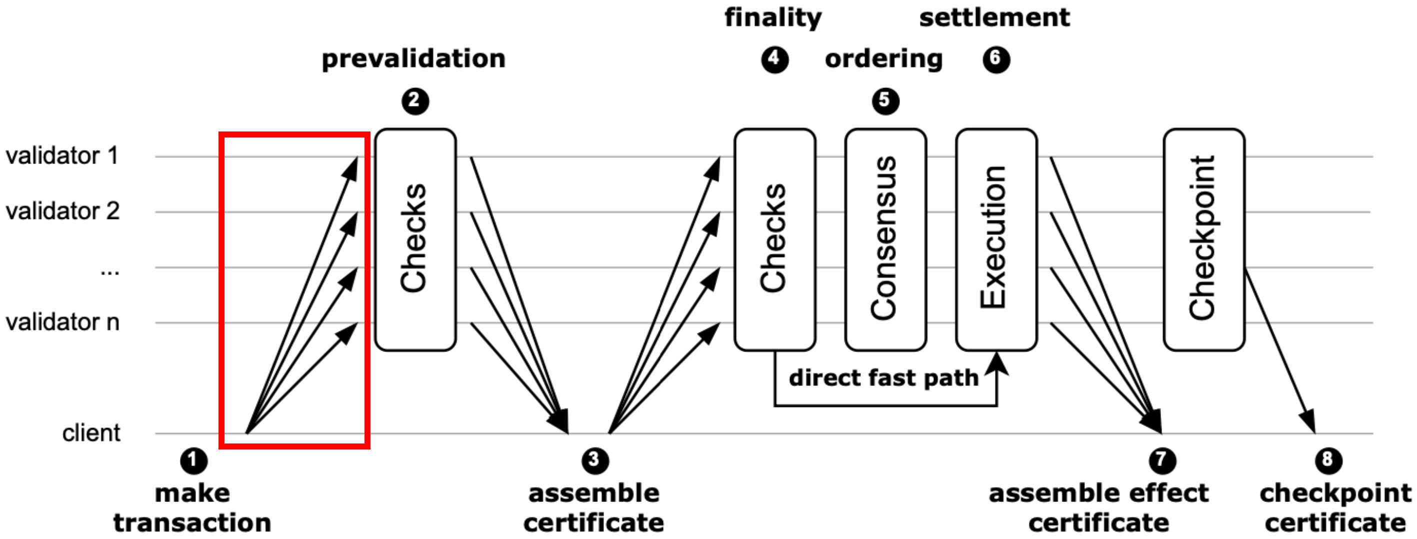
Write a: 10

Write b: a

Write a: 20

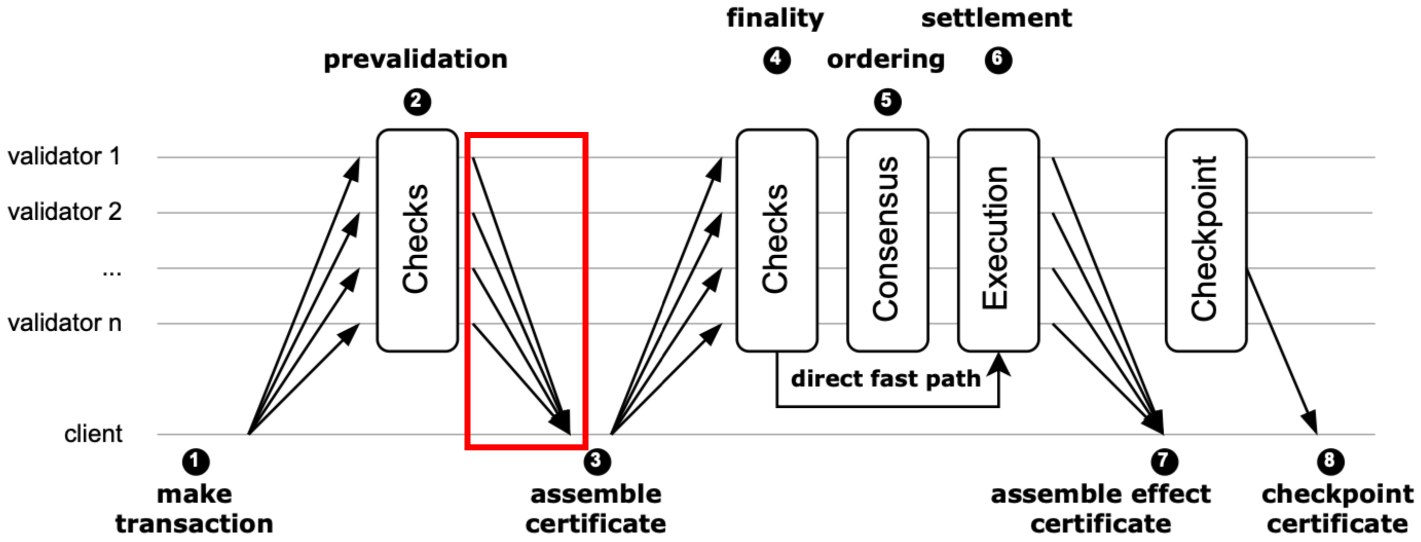
System Architecture

1. Client **broadcasts** its transaction to validators



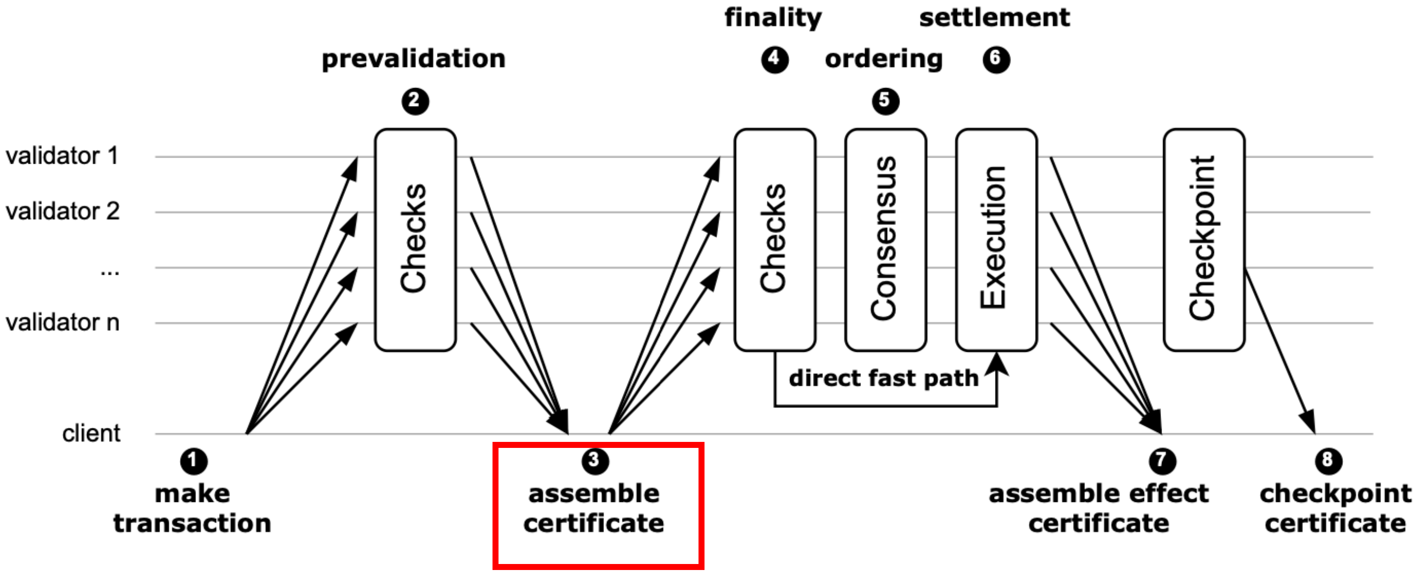
System Architecture

2. Validators perform validity checks and return the signed transaction to the client
    - Prevalidation (No execution)
    - Validators locally locks the input owned objects, using **ObjKey (Atomic test_and_set)**
    - Already locked ObjKey is involved -> **freeze** the object **for an epoch**



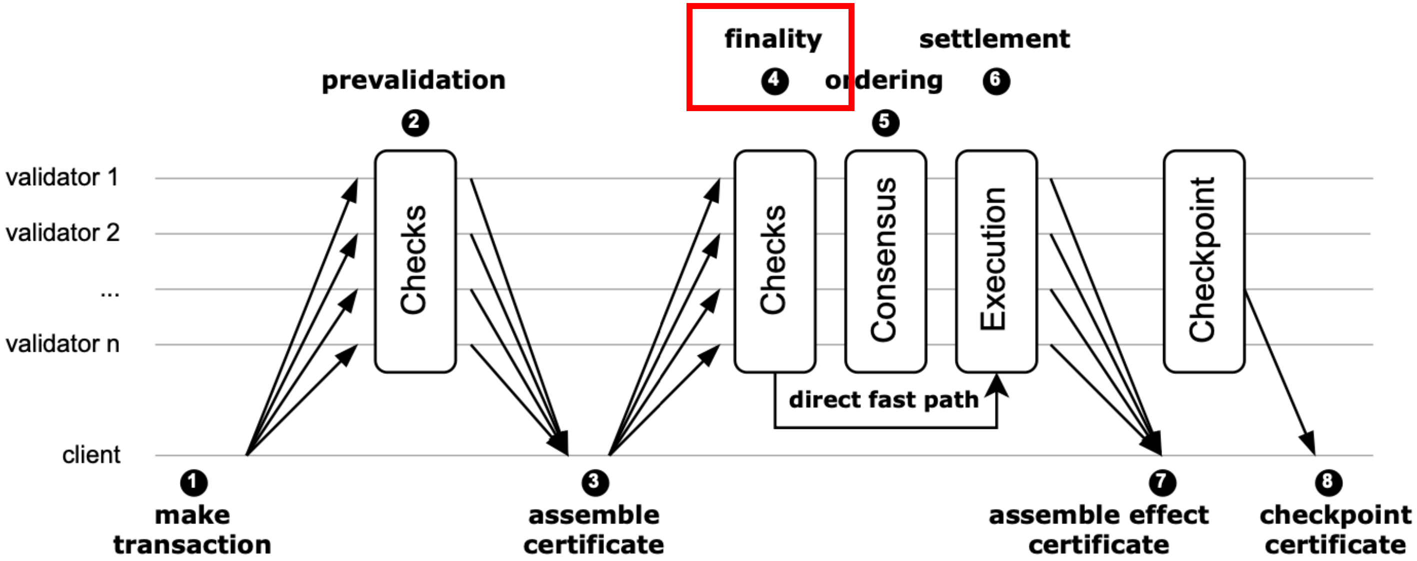
System Architecture

3. Client collects the responses from a *N-f* validators to form a **transaction certificate**
   - Signature aggregation algorithm (e.g. BLS)



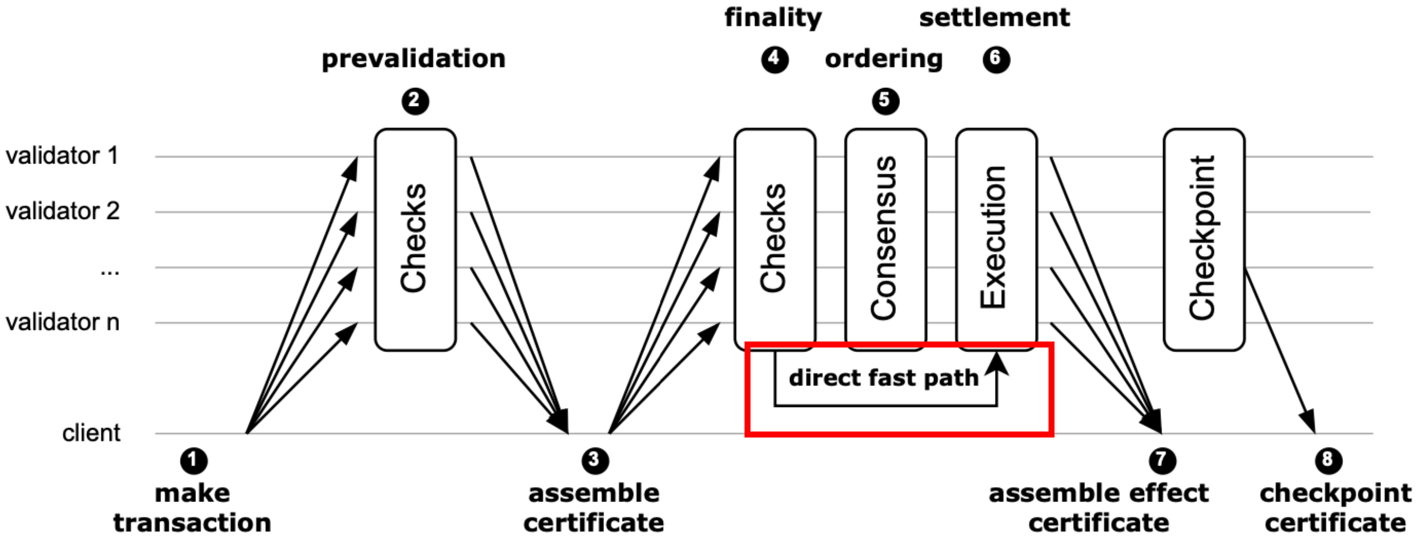
System Architecture

4. Broadcasts the certificate to validators
    - Check if the number of signers > *N*-*f*
    - **Transaction finality** (The execution of transaction is irrevokable)



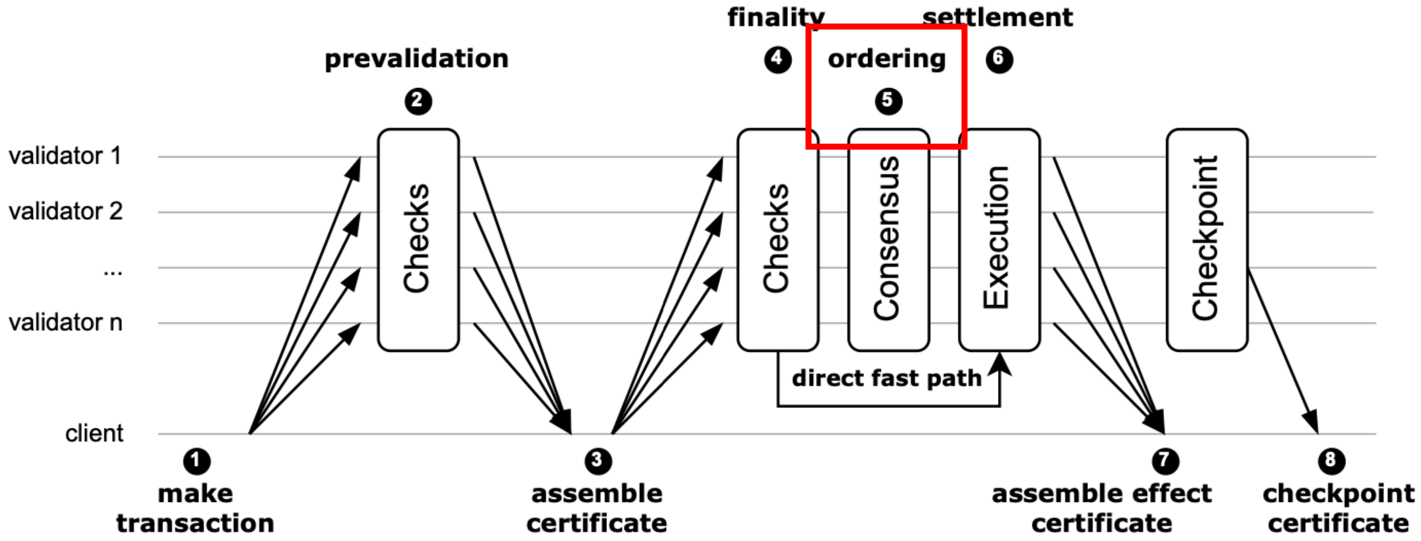
System Architecture

5. Transaction with certificate involves owned object → Execute without consensus
   - Skip ordering



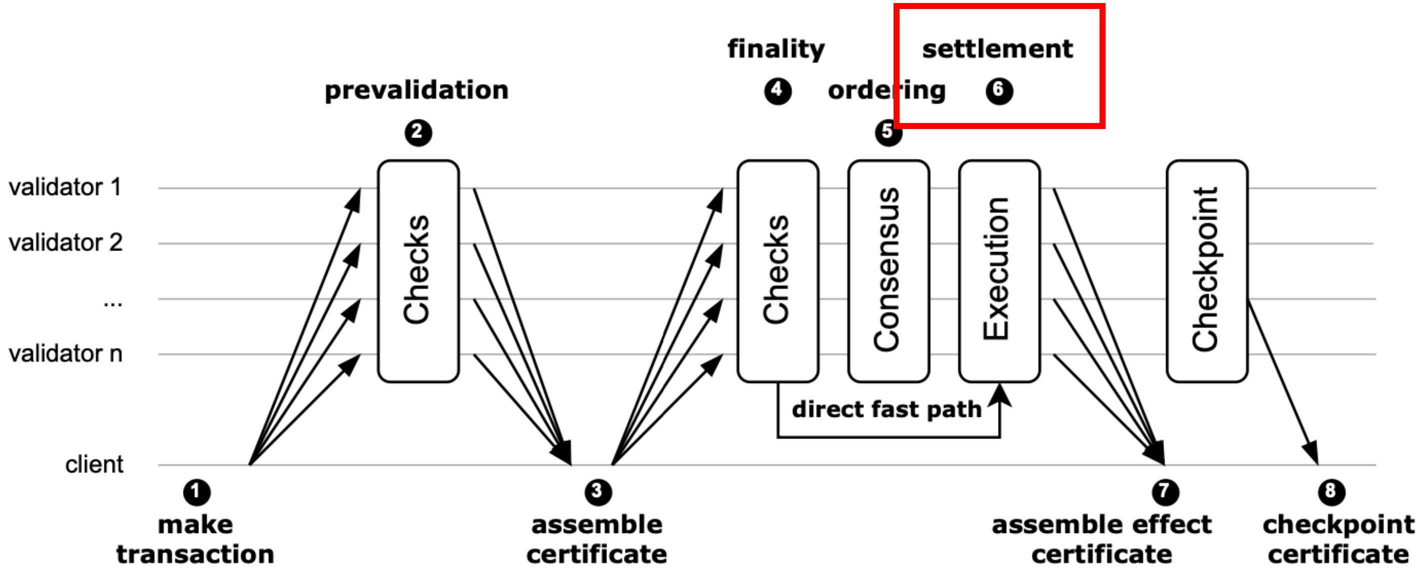
System Architecture

6. All certificates are forwarded to the consensus protocol
   - DAG-based Consensus (Blackbox)
   - Input: All certificates (Shared object + Owned object)
   - Output: Total order of certificates (Owned object cert first)



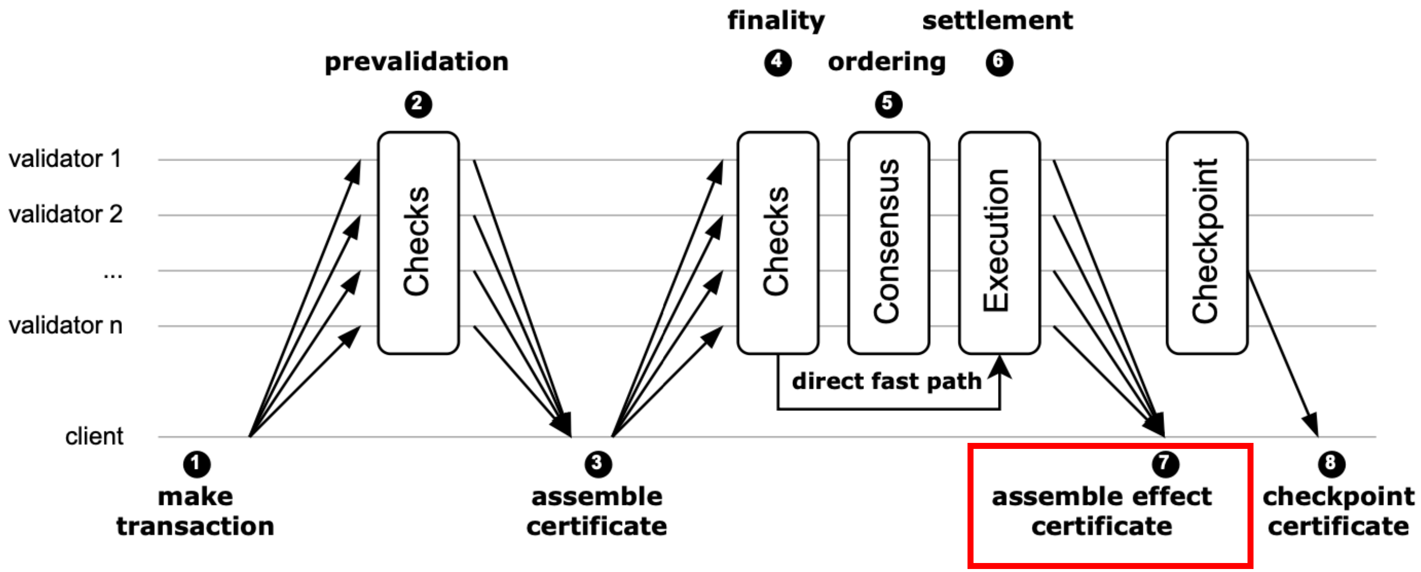
System Architecture

7. Validators execute transactions containing shared objects
   - Owned objects: Already executed by fast path
   - Settlement finality (The result of execution is irrevocable)



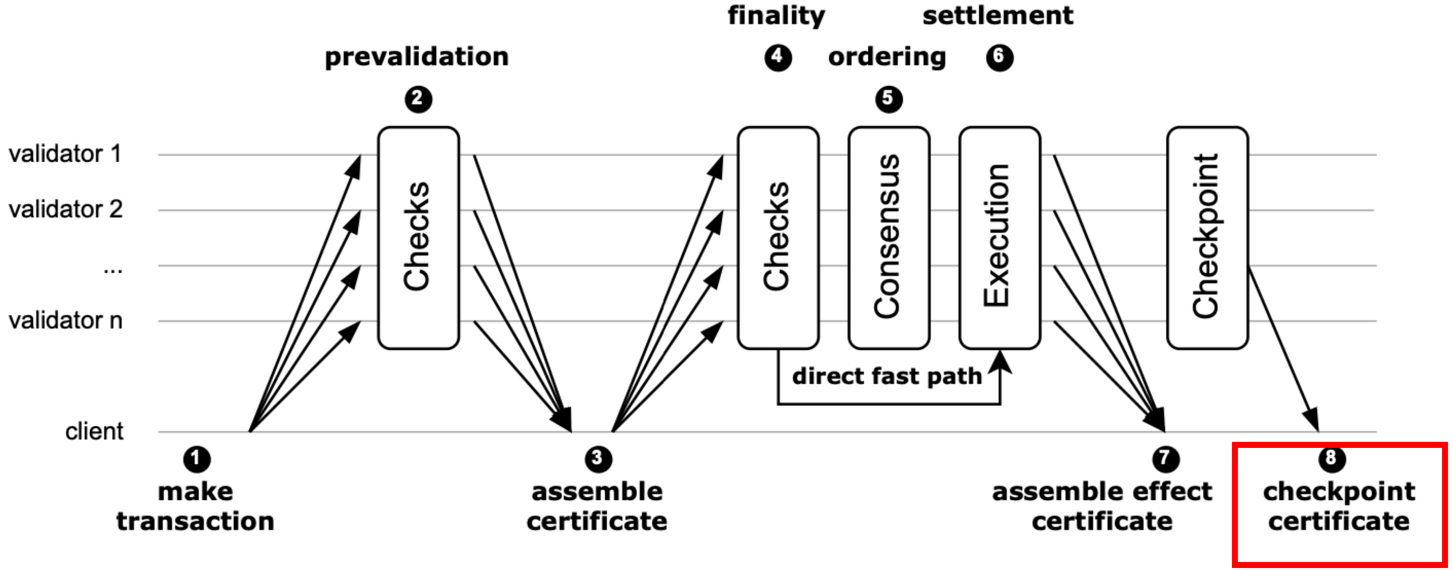
System Architecture

8. Client can collect N-f execution results and make effect certificate
    - Proof of settlement



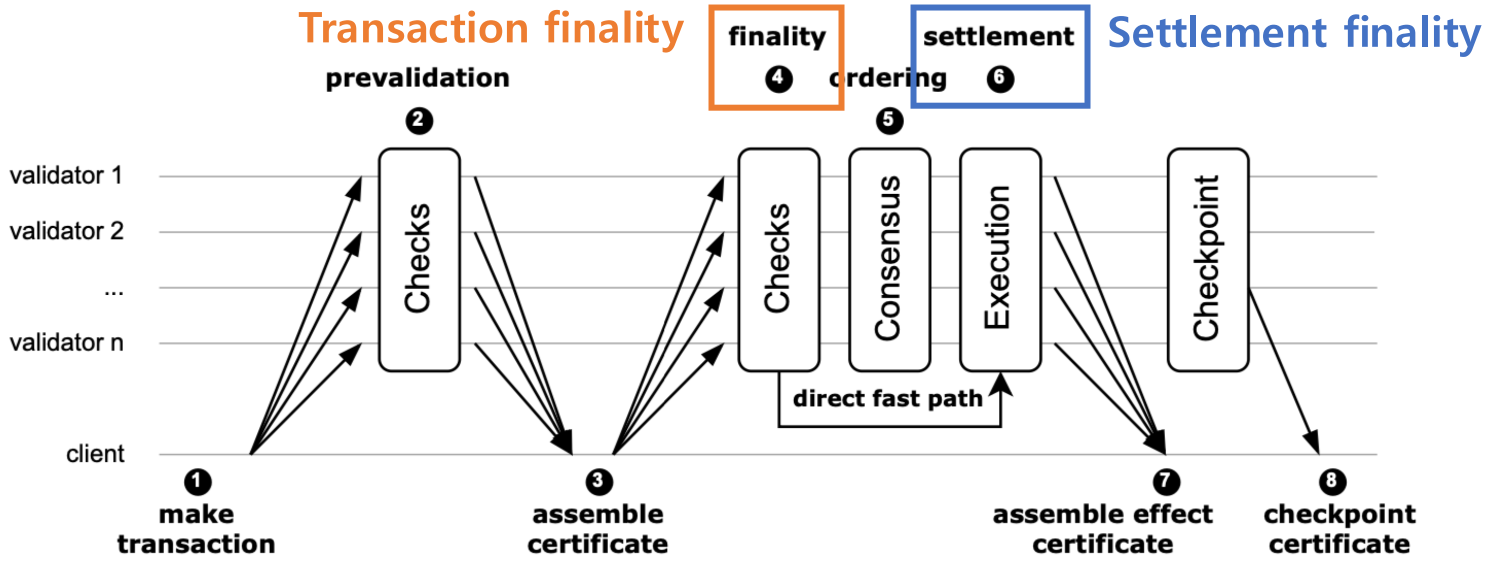
System Architecture

9. Create checkpoint based on the commit



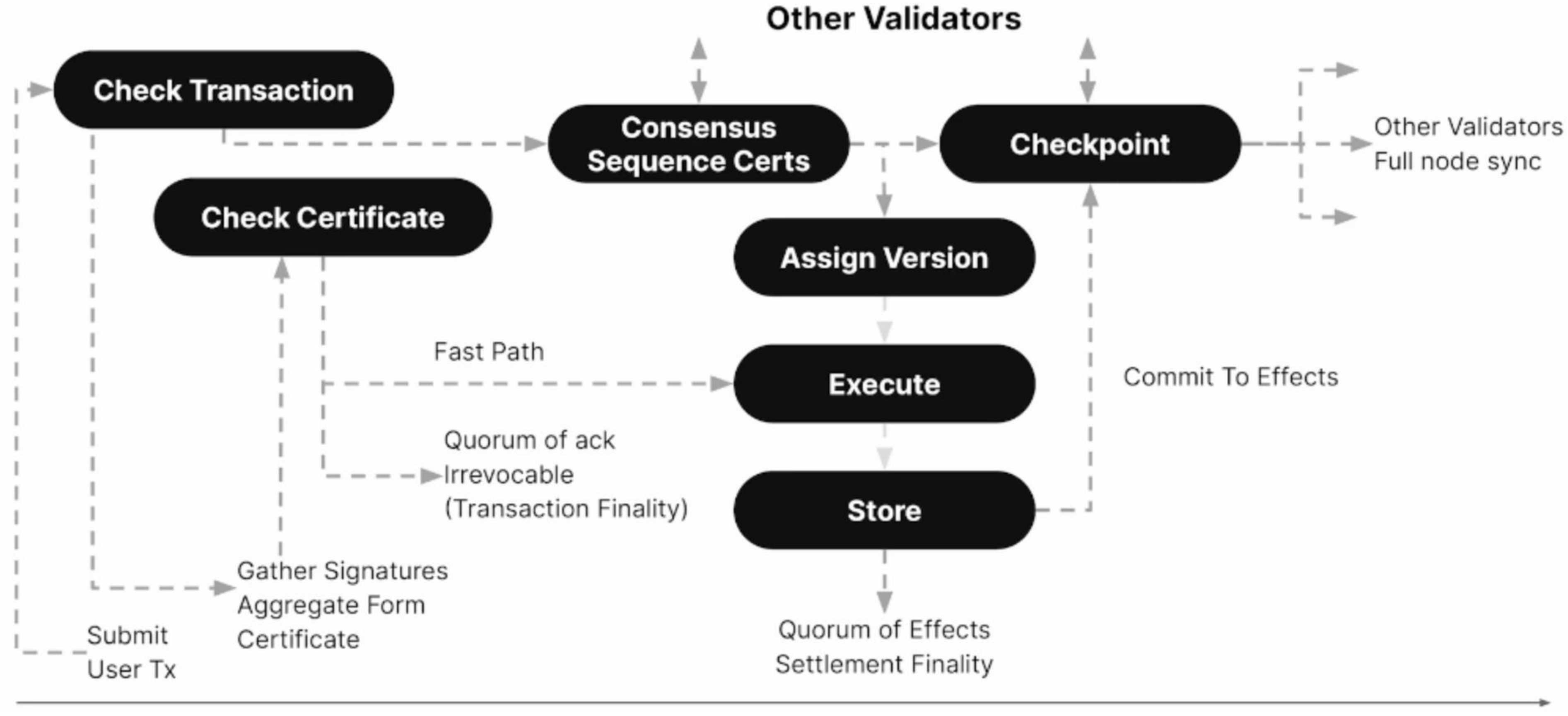
Two Types of Finality (Finality: irrevocable and unconditional)

- Transaction Finality
  - 트랜잭션의 실행이 final
  - 정직한 validator가 에포크 내에 트랜잭션을 실행할 것임이 보장됨
- Settlement Finality
  - 트랜잭션의 실행 결과가 final
  - 트랜잭션의 실행 결과를 subsequent transaction이 사용 가능
- Fast Path
  - Ordering을 우회하여 차이가 없음
- UX

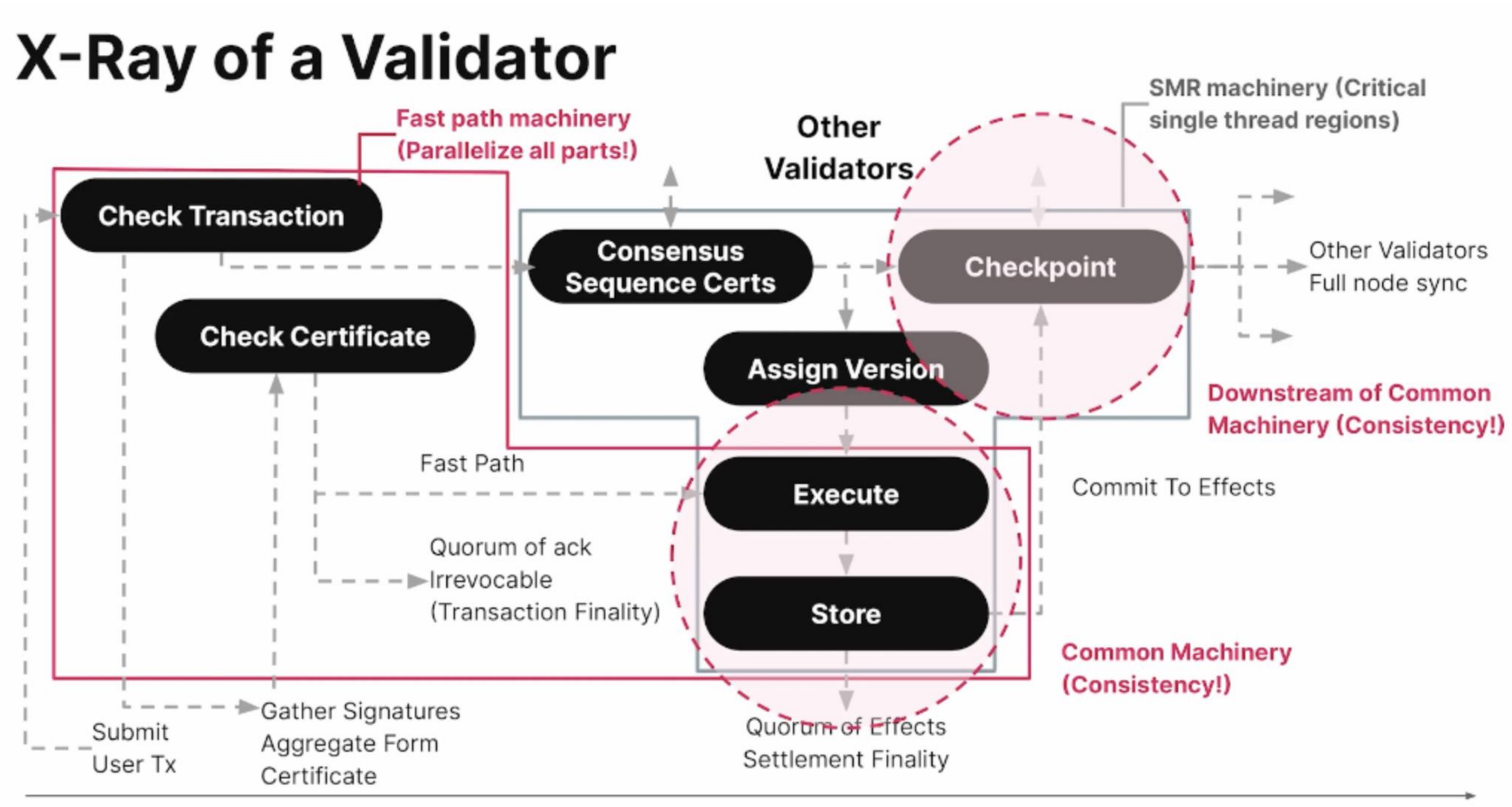


System Architecture

X-Ray of a Validator



System Architecture



X-Ray of a Validator

Fast path machinery (Parallelize all parts!)

Other Validators

SMR machinery (Critical single thread regions)

Check Transaction

Consensus Sequence Certs

Checkpoint

Other Validators Full node sync

Check Certificate

Assign Version

Downstream of Common Machinery (Consistency!)

Fast Path

Execute

Commit To Effects

Quorum of ack Irrevocable (Transaction Finality)

Store

Common Machinery (Consistency!)

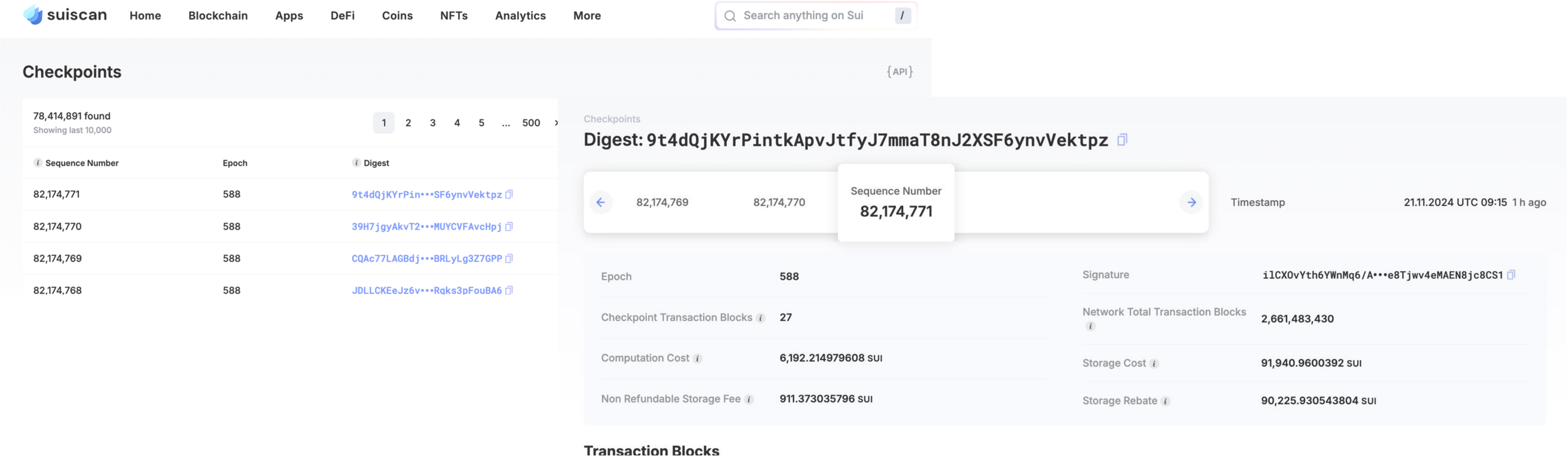
Submit User Tx

Gather Signatures Aggregate Form Certificate

Quorum of Effects Settlement Finality

Checkpoint

- Like the blocks of a traditional blockchain
  - Executed tx digests
  - Tx order
  - Signature of 2/3 committee



Checkpoints

78,414,891 found
Showing last 10,000

Sequence Number	Epoch	Digest
82,174,771	588	9t4dQjKYrPin•••SF6ynvVektpz
82,174,770	588	39H7jgyAkvT2•••MUYCVFAvcHpj
82,174,769	588	CQAc77LAGBdj•••BRLyLg3Z7GPP
82,174,768	588	JDLLCKEeJz6v•••Rqks3pFouBA6

Checkpoints

Digest: 9t4dQjKYrPintkApvJtfyJ7mmaT8nJ2XSF6ynvVektpz

82,174,769 82,174,770 Sequence Number 82,174,771

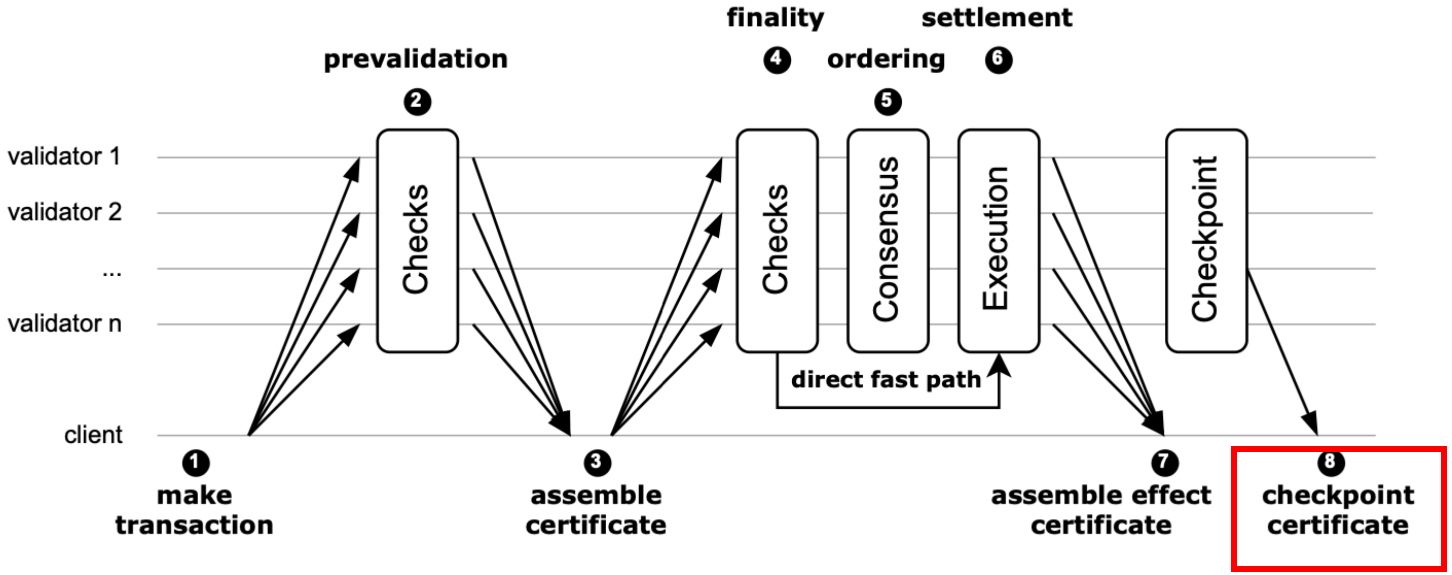
Timestamp 21.11.2024 UTC 09:15 1 h ago

Epoch	588	Signature	ilCXOvYth6YWnMq6/A•••e8Tjwv4eMAEN8jc8CS1
Checkpoint Transaction Blocks	27	Network Total Transaction Blocks	2,661,483,430
Computation Cost	6,192.214979608 SUI	Storage Cost	91,940.9600392 SUI
Non Refundable Storage Fee	911.373035796 SUI	Storage Rebate	90,225.930543804 SUI

Transaction Blocks

Checkpoint

- Periodically validators pick a consensus **commit** to use as a checkpoint
  - Current implementation: checkpoint per commit
  - 모든 certificate(owned + shared)에 대한 sequencing을 진행하기 때문에 가능



Reconfiguration

- Between epochs, when the current committee is replaced by a new committeee

1. Validators stop signing new transactions or lock objects
2. When all certificates executed locally are checkpointed, validator votes to close the epoch
3. If >2/3 validator vote, the epoch ends

- All freezes caused by equivocation will be dropped

Implementation

- Forked FastPay, Narwhal(Mempool), Bullshark(Consensus)
- RocksDB
- QUIC

- Production level
  - Sui mainnet
  - 107 geo-distributed validators
  - 3.1 million certificate per day

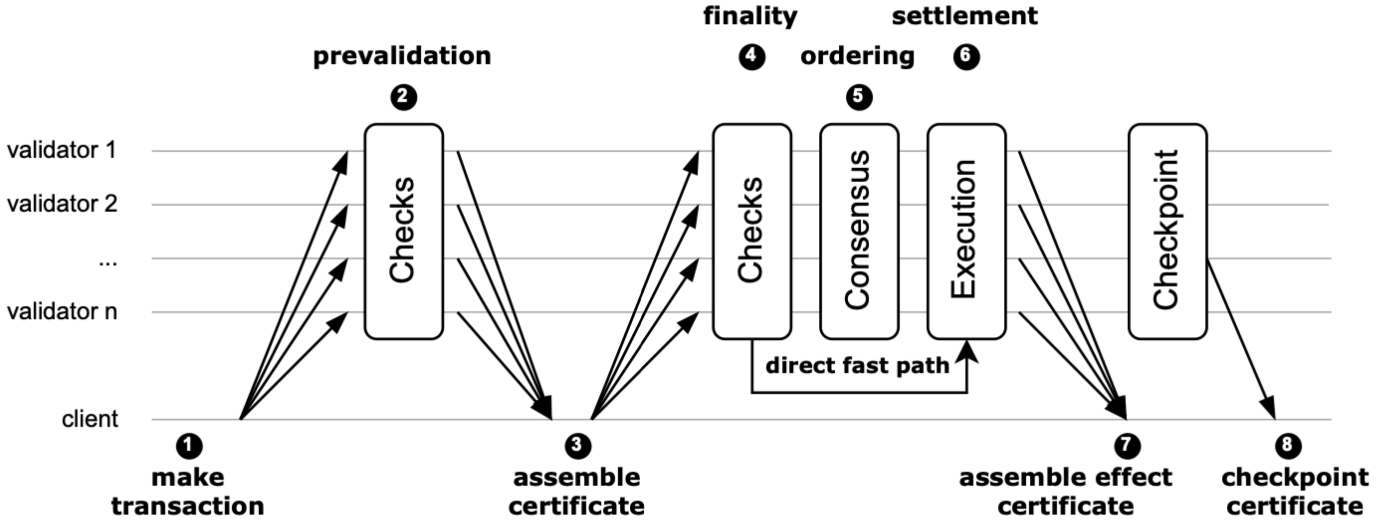
Experiment Setup

- AWS m5d.8xlarge instances
  - 10Gbps bandwidth
  - 2.5GHz 32 virtual CPUs (16 physical core)
  - 128GB memory
  - Ubuntu 22.04
- Commodity servers
- 13 different AWS regions(Virginia, Oregon, Canada, Frankfurt, Ireland, London, Paris, Stockholm, Mumbai, Singapore, Sydney, Tokyo, Seoul)

Experiment Setup

- 350 tx/s for 10 minutes
- Latency : Settlement finality – Tx submission time
- Throughput : The number of effect certificates

- Common Case(No Fault) / Faults
- Baseline: Bullshark
  - Extended version
  - Sui lutris without FastPay
- Tx: payment + contract call



Benchmark with Common Case

- Sub-second latency
  - X6 improvement
- Better Throughput
  - Regardless of committee size

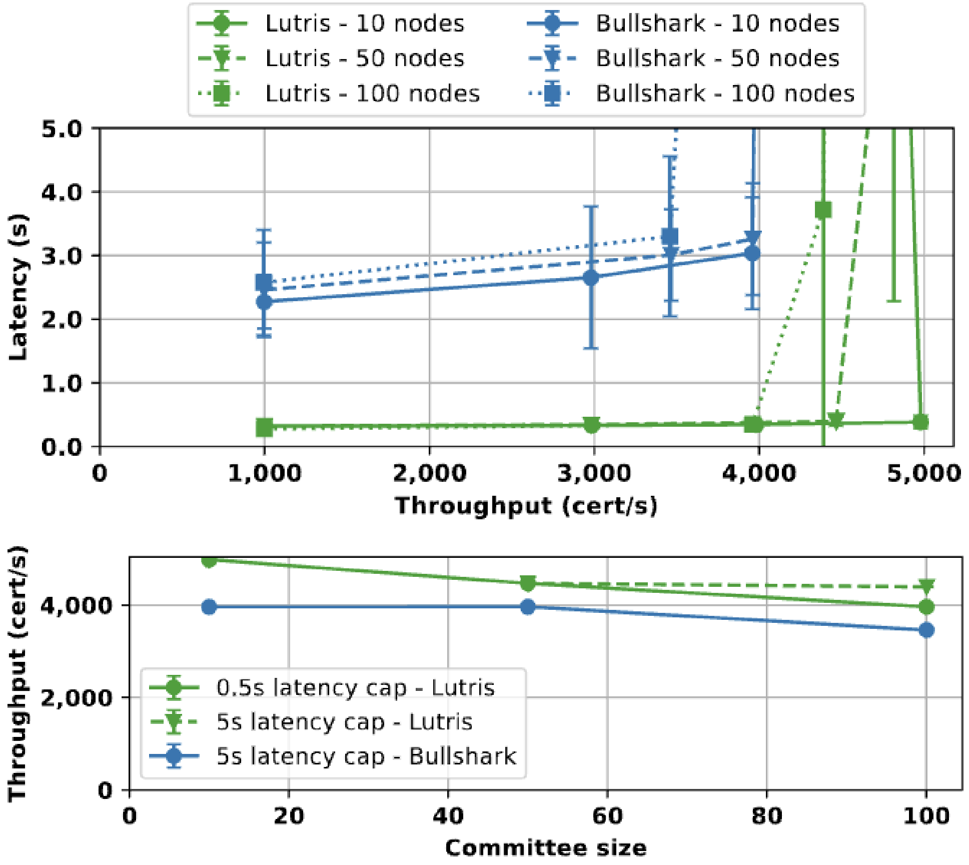


Figure 4: Sui Lutris and Bullshark WAN latency-throughput with 10, 50, and 100 validators (no faults).

Benchmark with Common Case

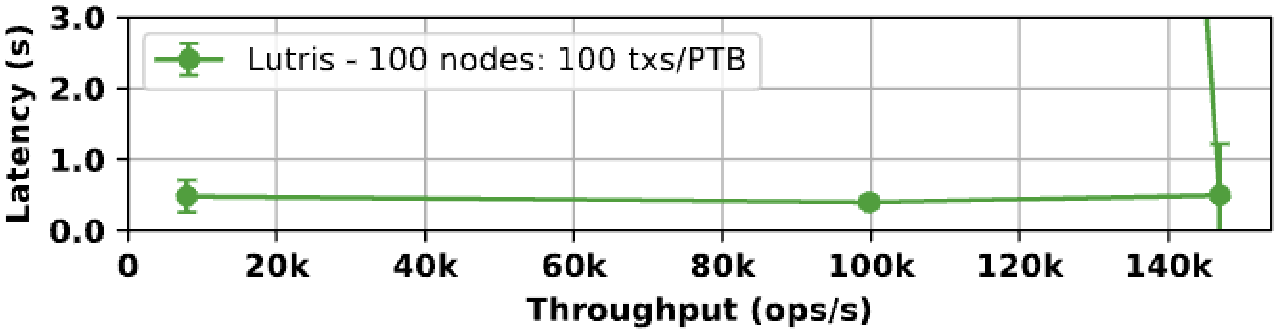


Figure 5: Sui Lutris latency-throughput with bundles of 100 transactions per programmable transaction block (PTB); 100 validators, no faults.

- PTB: 트랜잭션 번들
- Owned PTB : Shared PTB == 60: 40
- 최대 150,000 ops/s
  - 1500 PTB

Benchmark with Faults

- 최대 15배 latency
  - Lutris: 0.5 seconds, 4000 cert/s
  - Bullshark
    - 1 faulty node: 5 seconds, 3500 cert/s
    - 3 faulty nodes: 7.5 seconds, 3000 cert/s

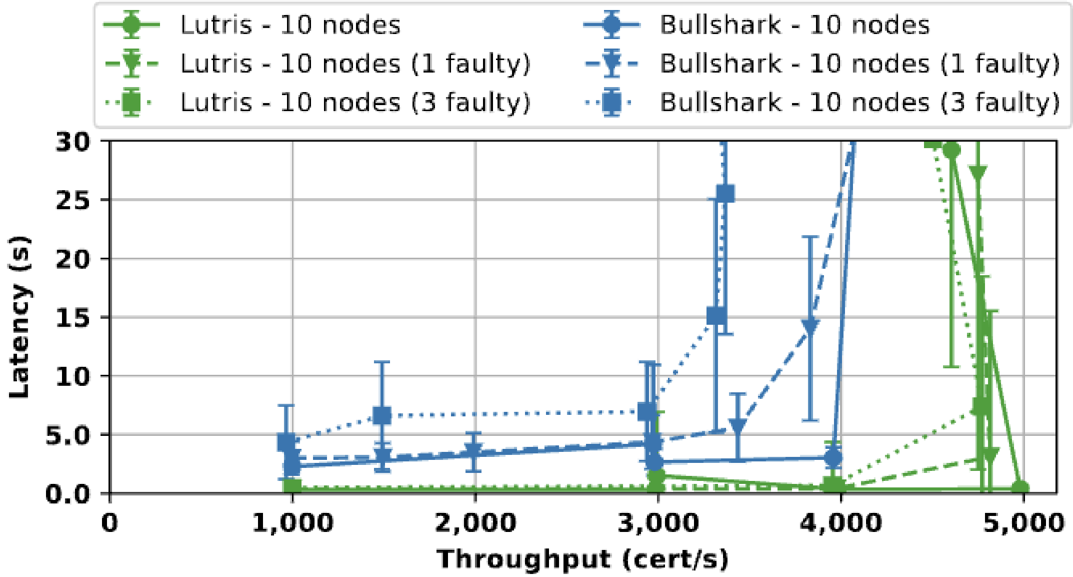


Figure 6: Sui Lutris and Bullshark WAN latency-throughput with 20 validators (1, 3, and 6 faults).

Benchmark with Faults

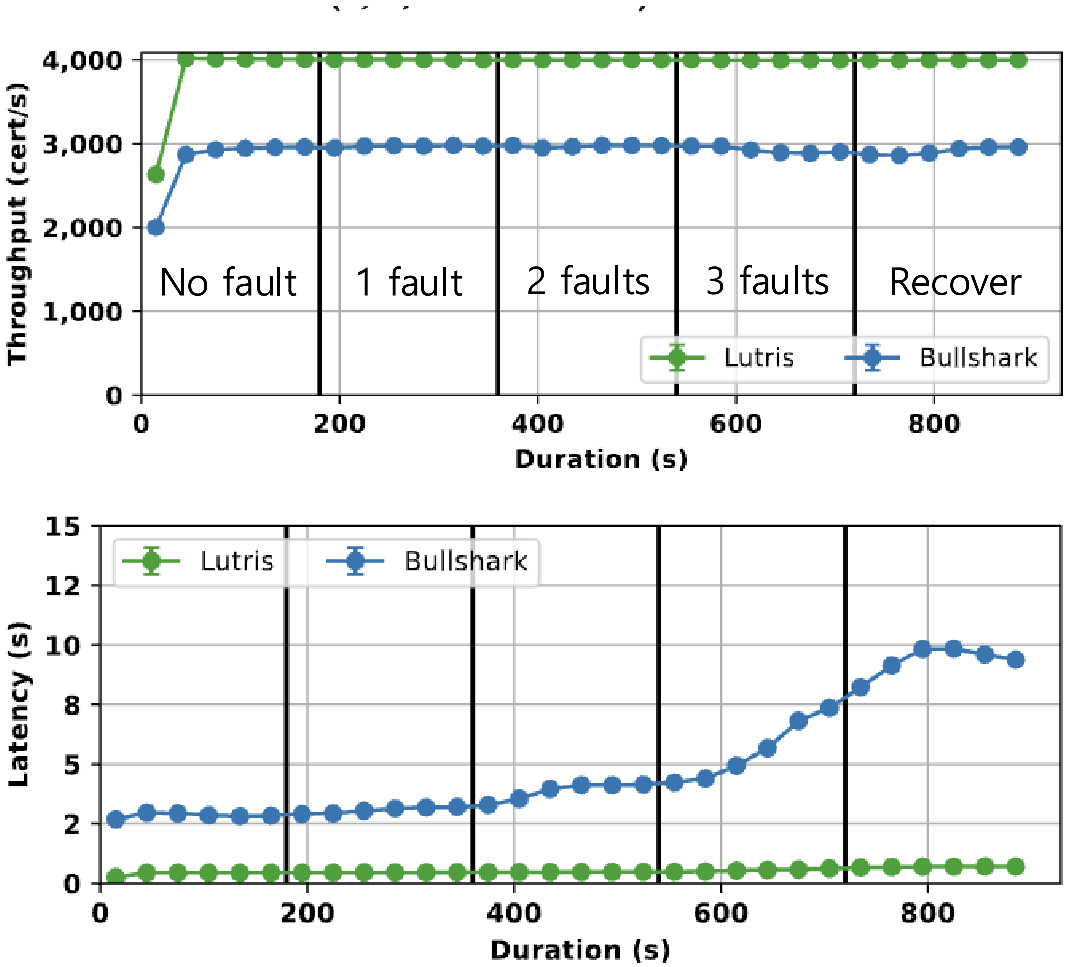


Figure 7: Performance of a 10-validators committee when up to 3 validators crash and recover.