Security Concerns and Vulnerabilities in Blockchain Systems

Prof. Gordon Pace
Malta Digital Innovation Authority
September 2019

Security Concerns and Vulnerabilities in Blockchain Systems

- **Security concerns at the blockchain layer:**
  - blockchain node bugs
  - consensus algorithms and vulnerabilities (50%+1 attack, forking, rejected blocks).
- **Security concerns at the smart contract logic layer:**
  - bugs and vulnerabilities in the smart contract logic;
  - malicious code;
  - vulnerabilities at the blockchain level (e.g. transaction ordering)
- **Security concerns at the edge of the blockchain:**
  - traditional vulnerabilities through off-chain access of blockchain and smart contract
  - human factors in security concerns.

Security Concerns and Vulnerabilities in Blockchain Systems

- **Security c**
  - blockc
  - conse
- **Security c**
  - bugs a
  - malici
  - vulner
- **Security c**
  - traditional vulnerabilities through off-chain access of blockchain and smart contract
  - human factors in security concerns.

cted blocks).

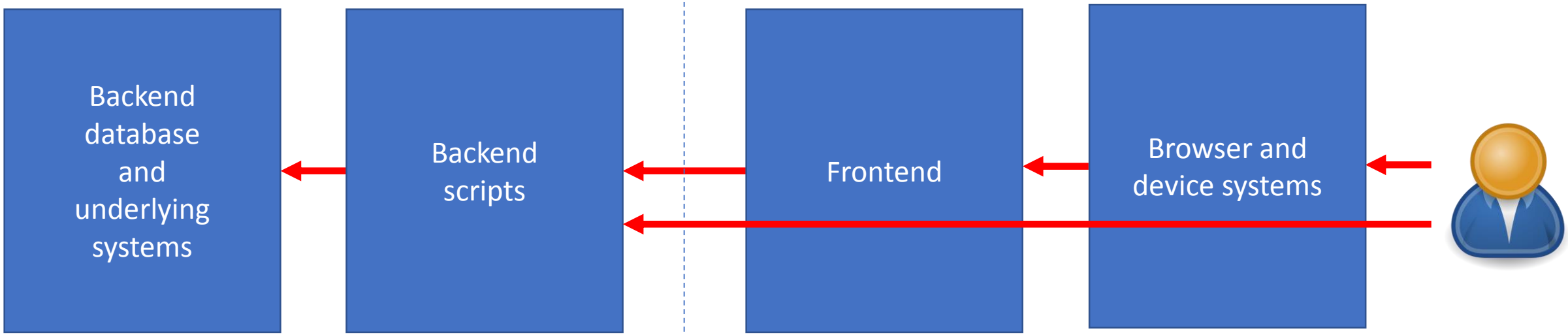
Undesirable human behaviour
using expected application functionality
vs.
Undesirable human behaviour
due to unexpected application functionality

Security Concerns and Vulnerabilities in Blockchain Systems

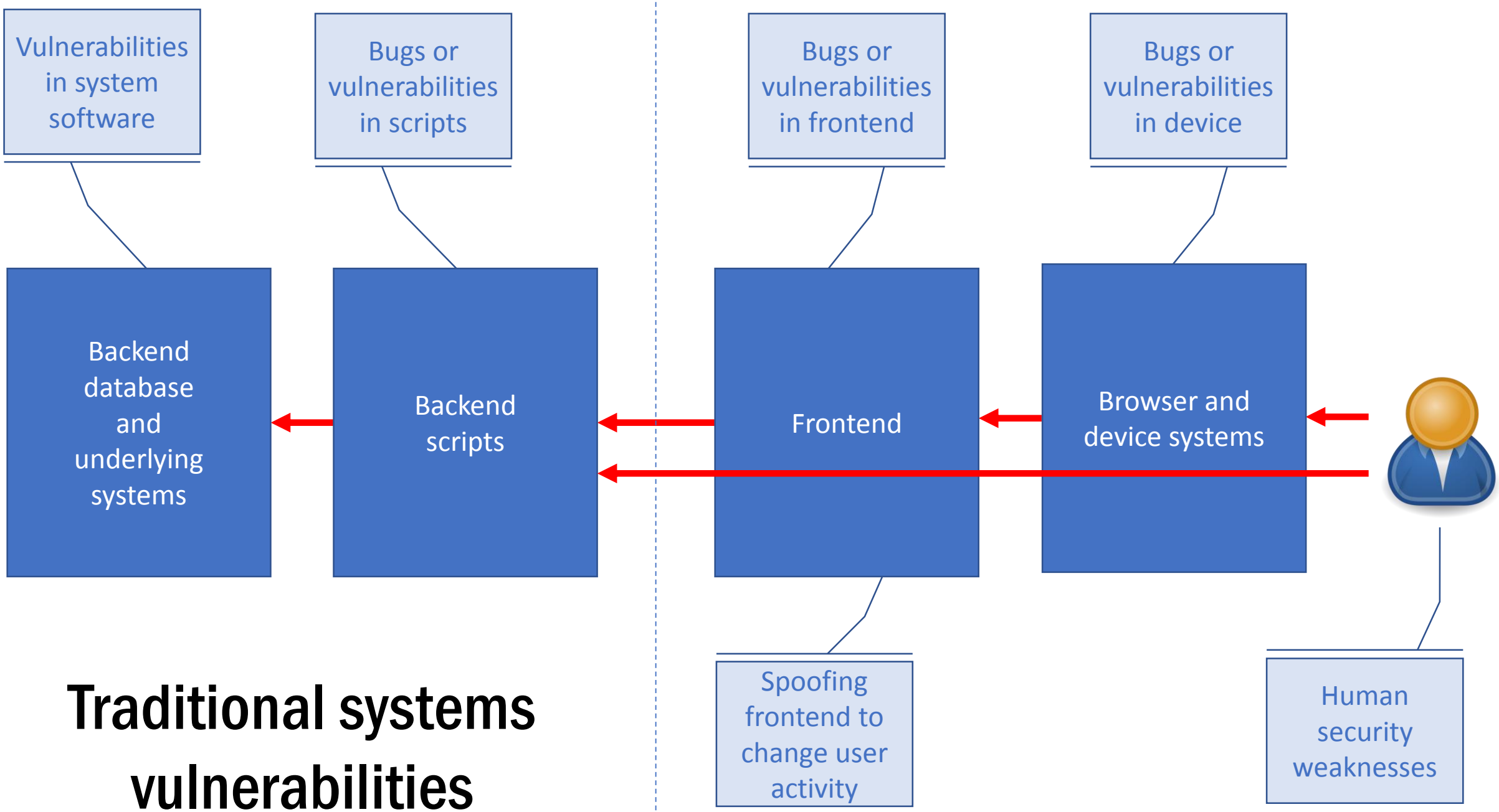
- **Security c**
  - blockc
  - conse
- **Security c**
  - bugs a
  - malici
  - vulner
- **Security c**
  - traditional vulnerabilities through off-chain access of blockchain and smart contract
  - human factors in security concerns.

cted blocks).

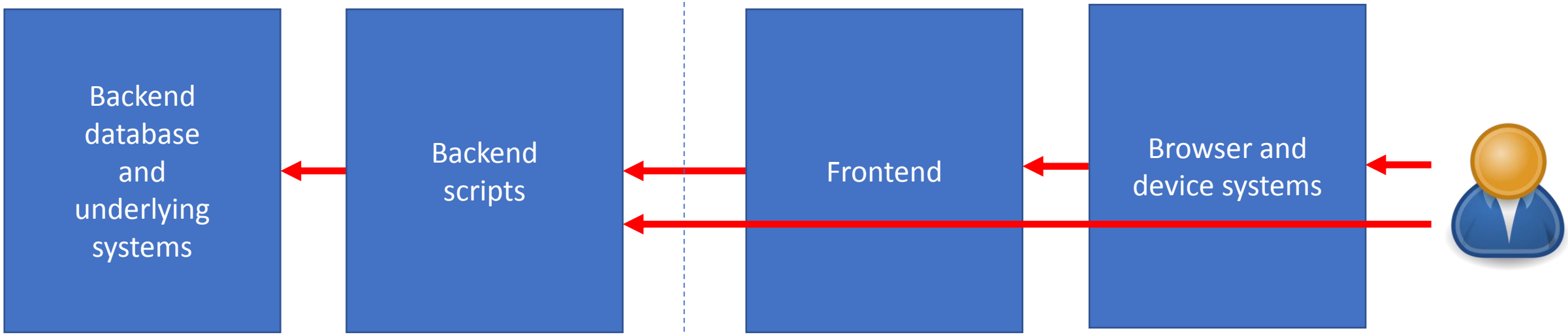
~~Undesirable human behaviour~~
~~using expected application functionality~~
vs.
Undesirable human behaviour
due to unexpected application functionality



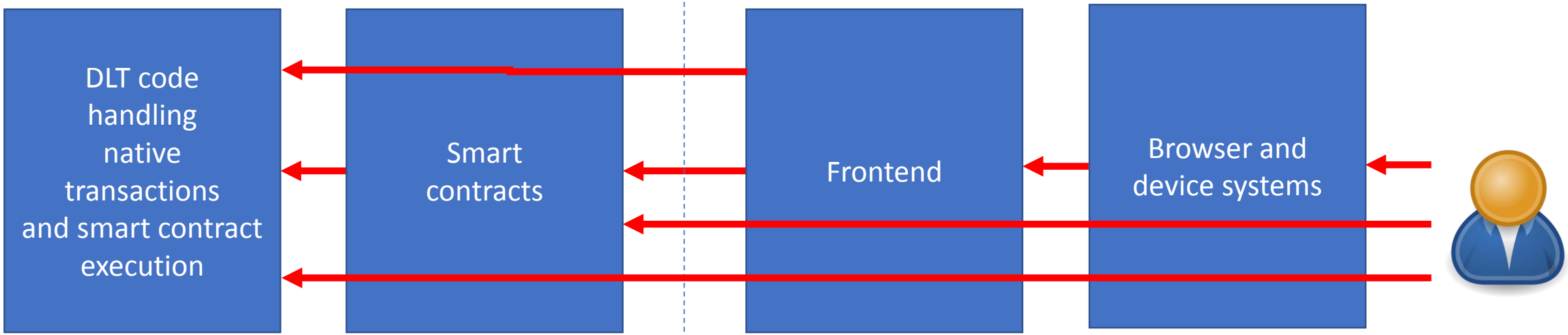
Traditional systems



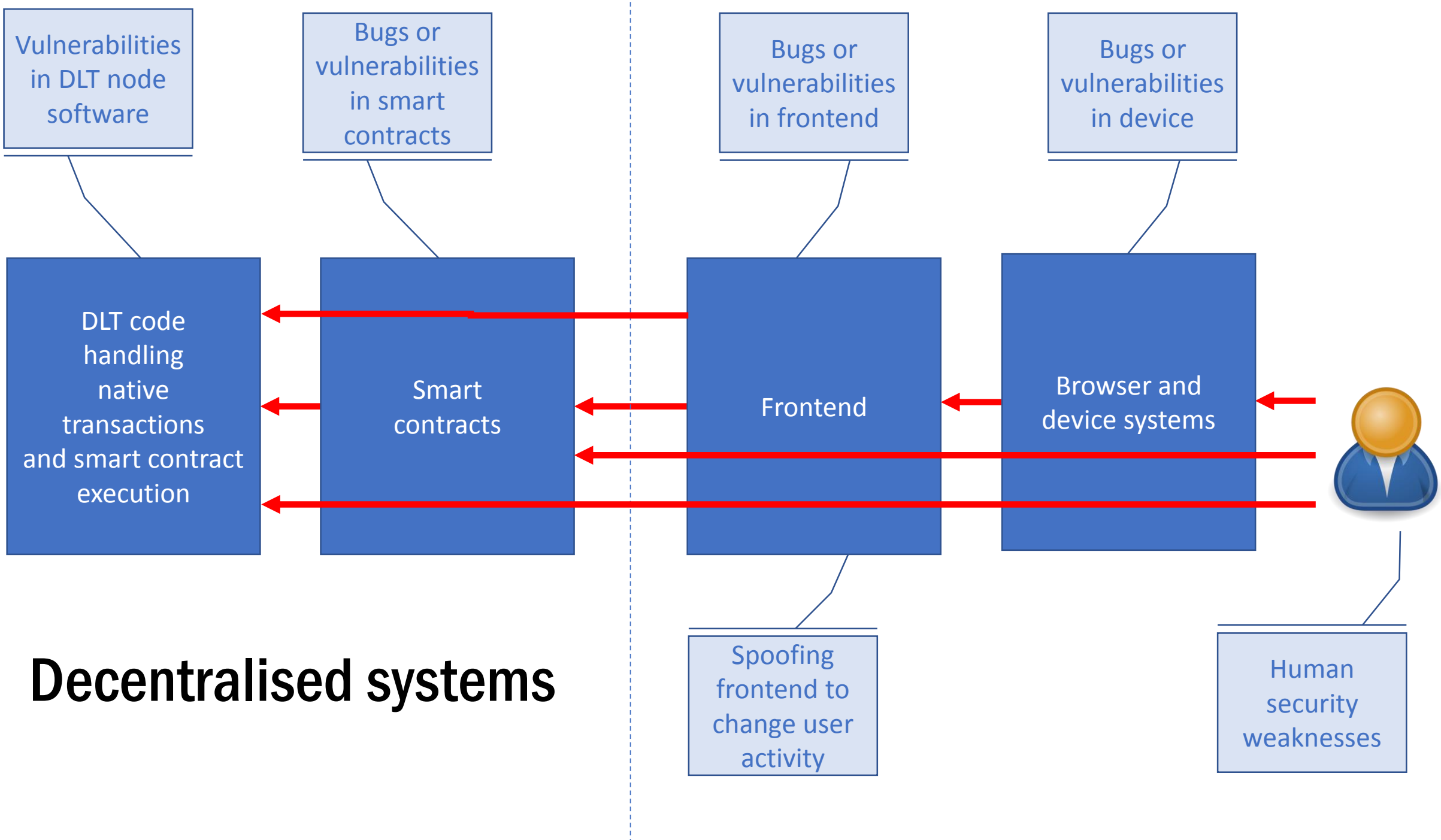
Traditional systems vulnerabilities



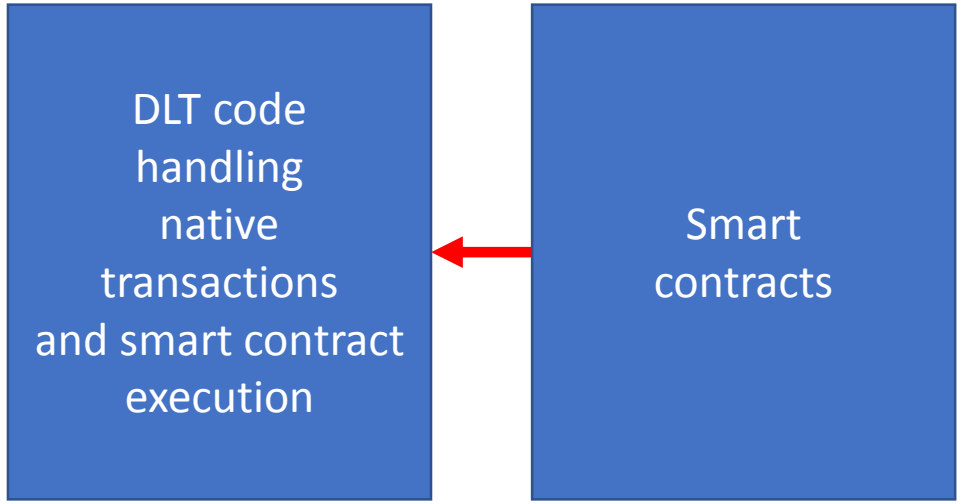
Traditional systems

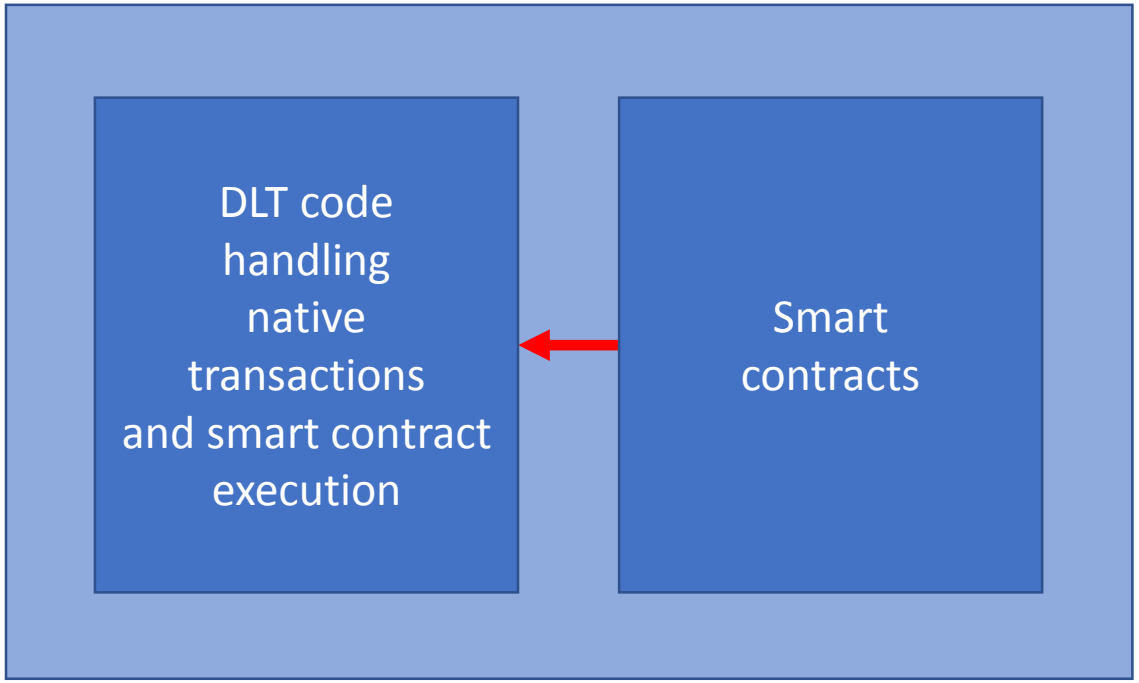


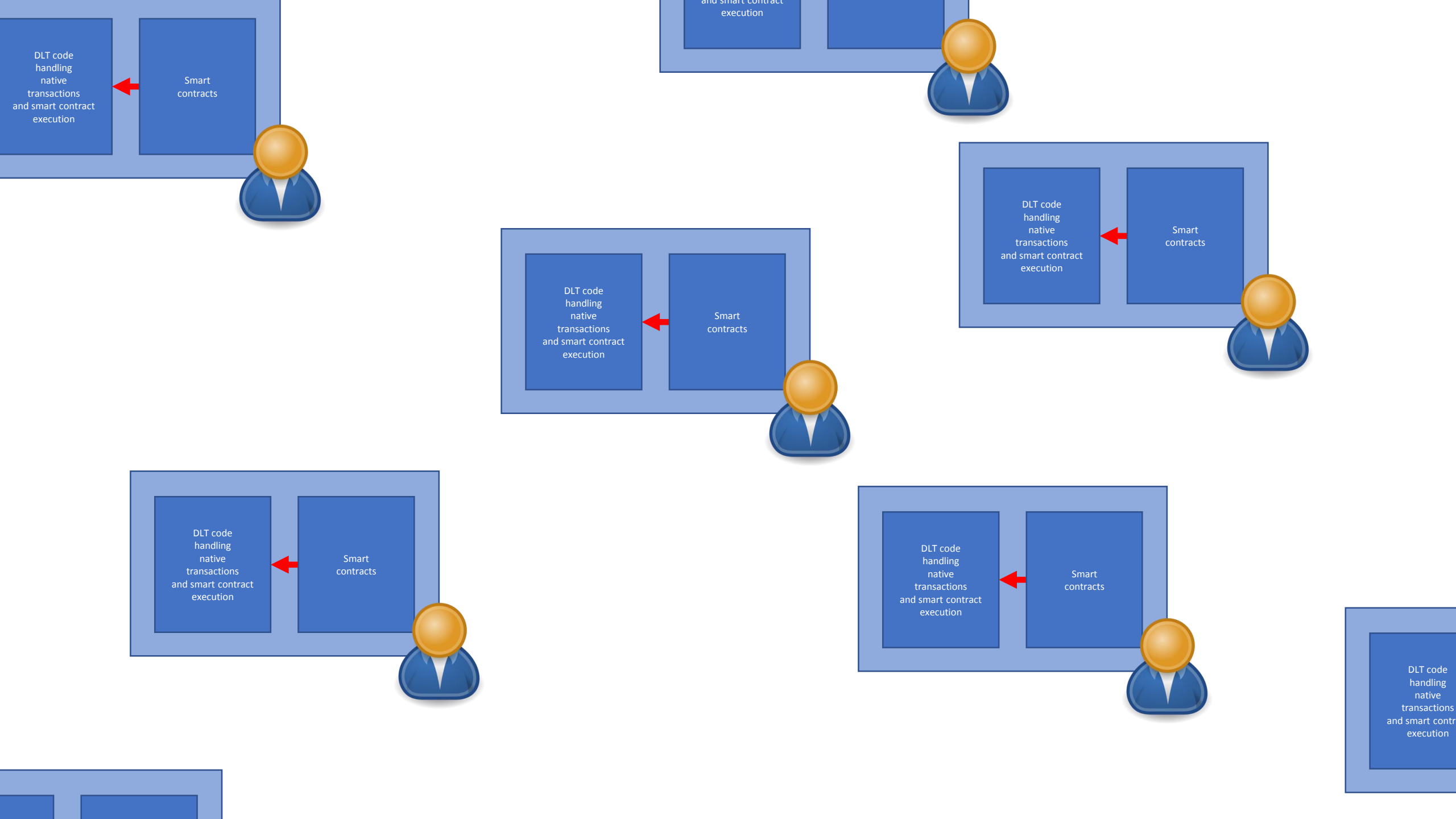
Decentralised systems

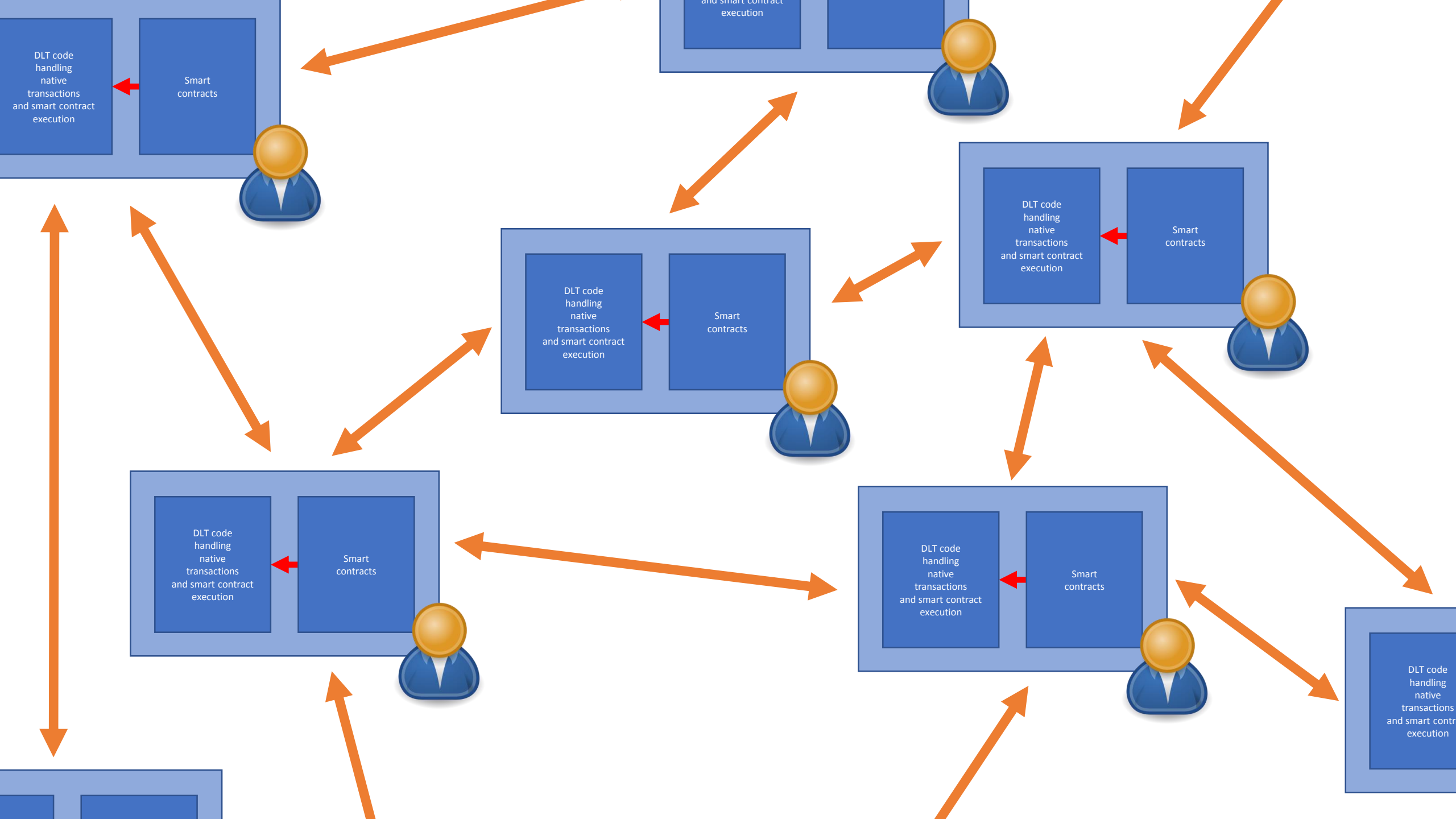


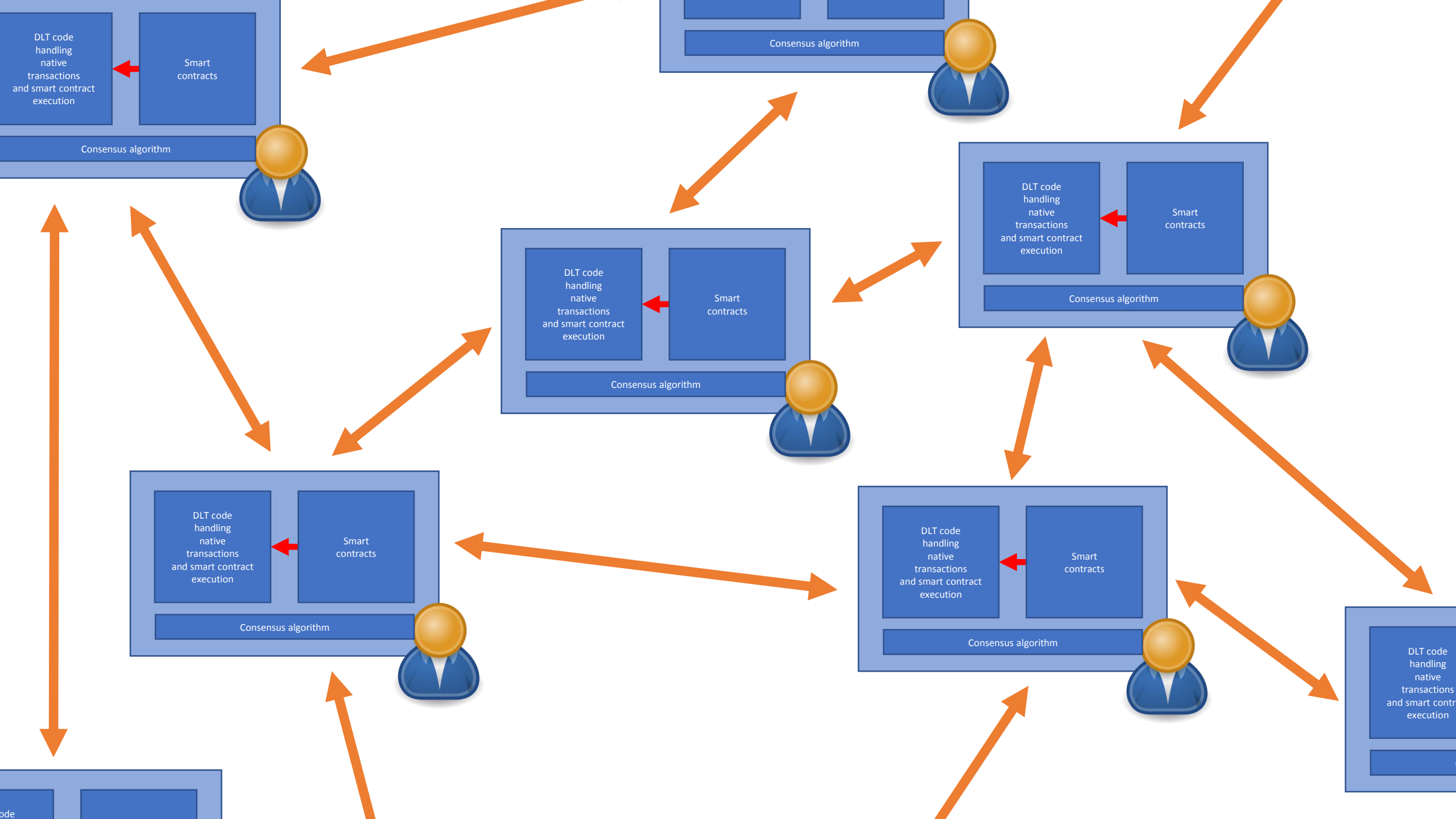
Decentralised systems

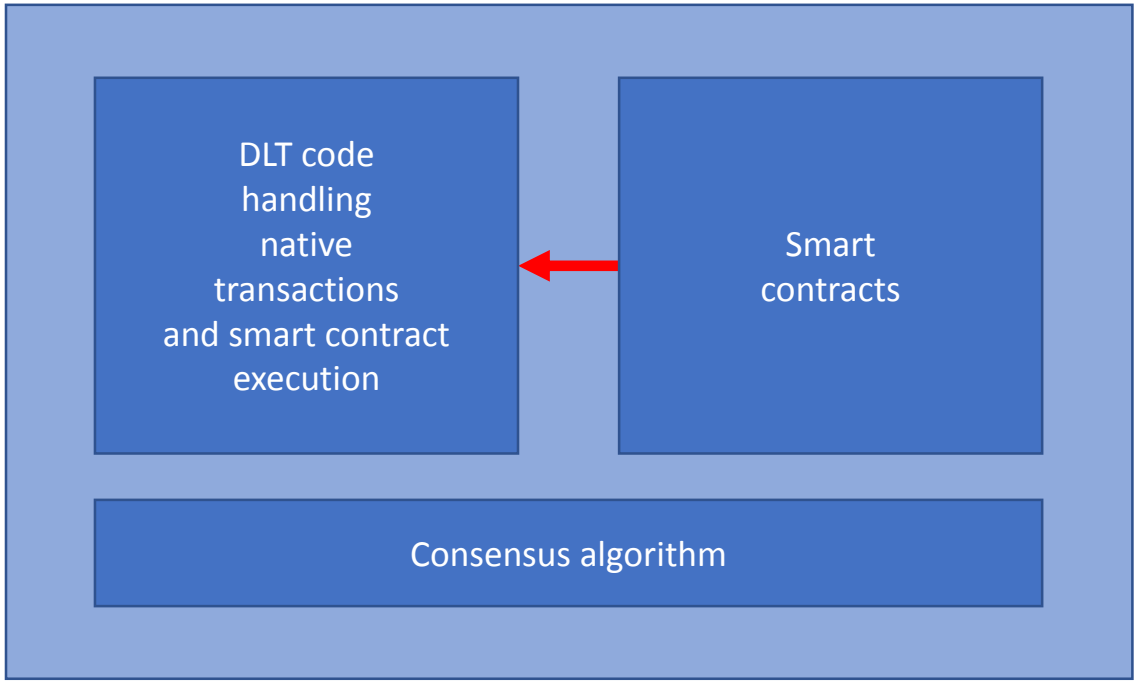

DLT code handling native transactions and smart contract execution
Smart contracts

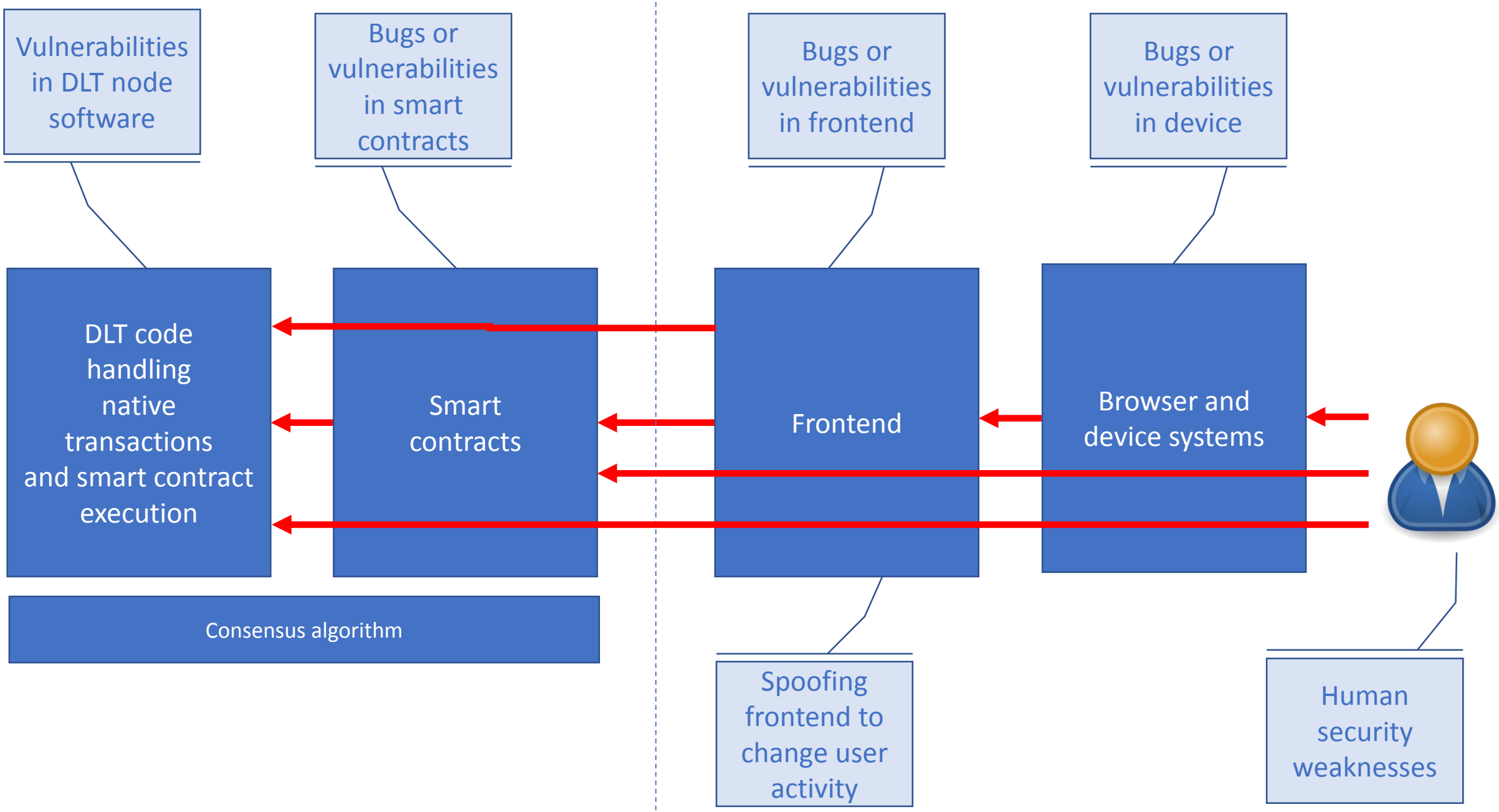

DLT code handling native transactions and smart contract execution
Smart contracts

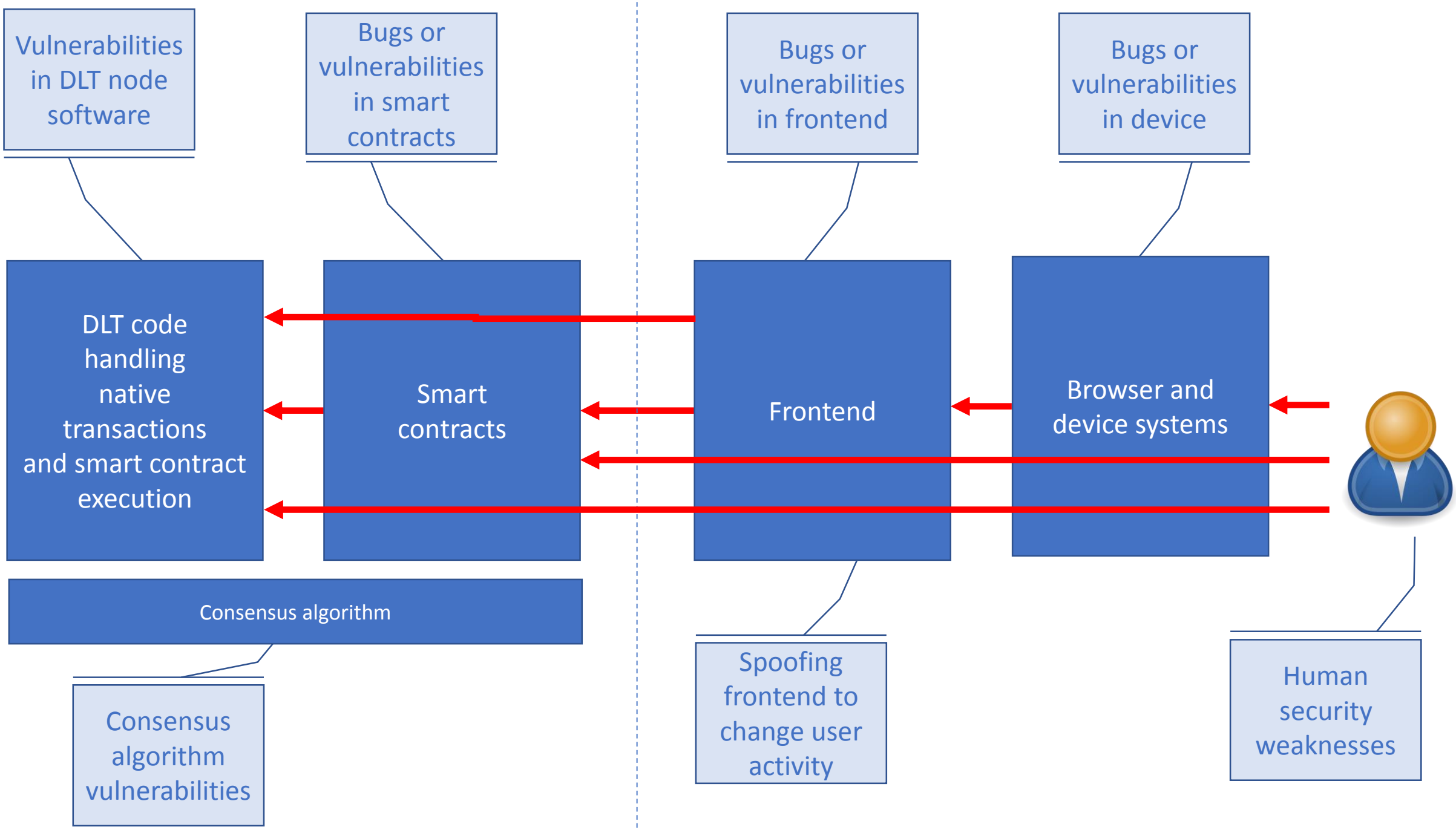

DLT code handling native transactions and smart contract execution
Smart contracts
DLT code handling native transactions and smart contract execution
Smart contracts
DLT code handling native transactions and smart contract execution
Smart contracts
DLT code handling native transactions and smart contract execution
Smart contracts
DLT code handling native transactions and smart contract execution
Smart contracts
DLT code handling native transactions and smart contr execution

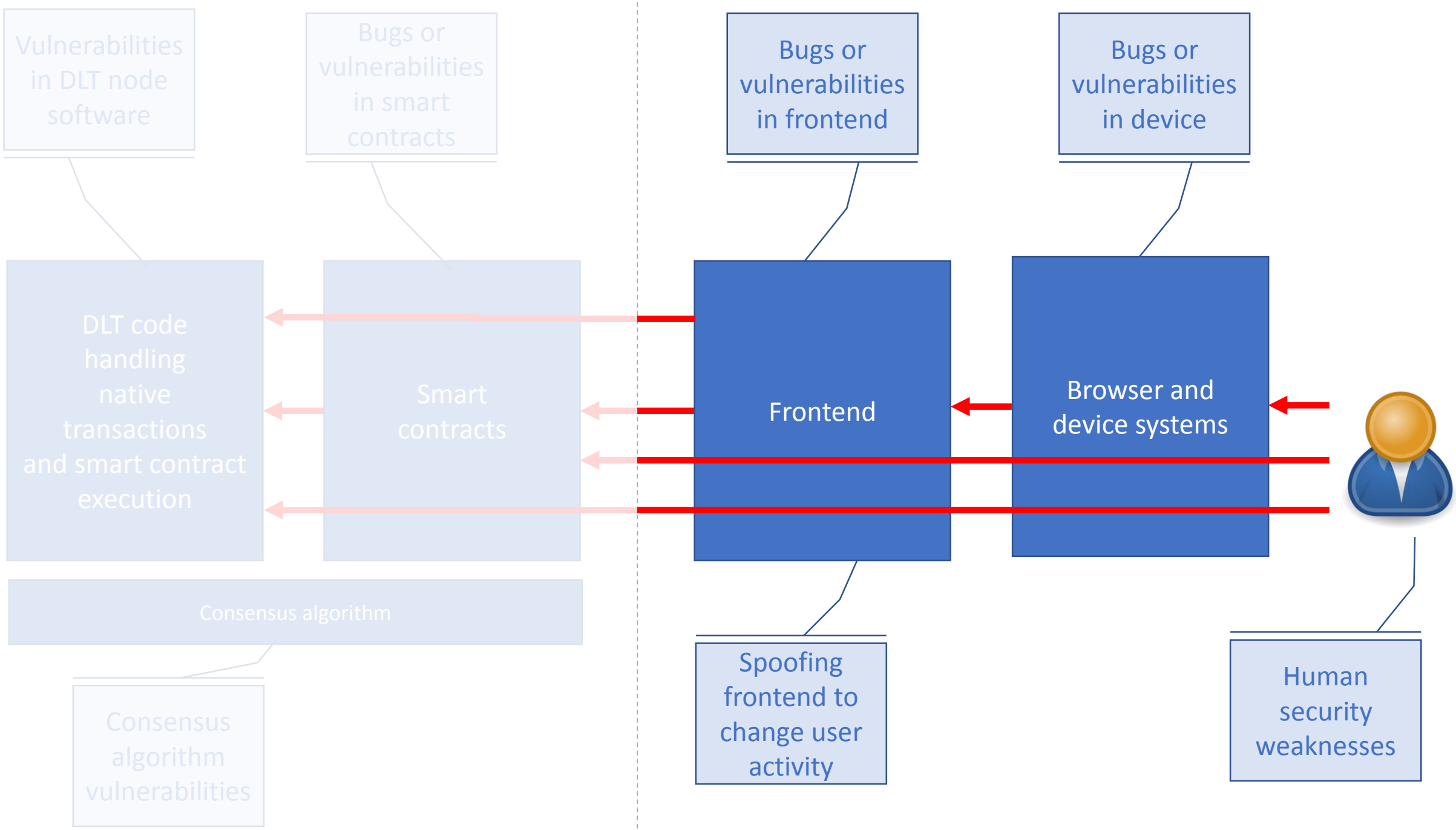

DLT code handling native transactions and smart contract execution
Smart contracts
DLT code handling native transactions and smart contract execution
Smart contracts
DLT code handling native transactions and smart contract execution
Smart contracts
DLT code handling native transactions and smart contract execution
Smart contracts
DLT code handling native transactions and smart contract execution
Smart contracts
DLT code handling native transactions and smart contr execution


DLT code handling native transactions and smart contract execution
Smart contracts
Consensus algorithm
DLT code handling native transactions and smart contract execution
Smart contracts
Consensus algorithm
DLT code handling native transactions and smart contract execution
Smart contracts
Consensus algorithm
DLT code handling native transactions and smart contract execution
Smart contracts
Consensus algorithm
DLT code handling native transactions and smart contract execution
Smart contracts
Consensus algorithm
DLT code handling native transactions and smart contr
execution


DLT code handling native transactions and smart contract execution
Smart contracts
Consensus algorithm


Vulnerabilities in DLT node software
Bugs or vulnerabilities in smart contracts
Bugs or vulnerabilities in frontend
Bugs or vulnerabilities in device
DLT code handling native transactions and smart contract execution
Smart contracts
Frontend
Browser and device systems
Consensus algorithm
Spoofing frontend to change user activity
Human security weaknesses

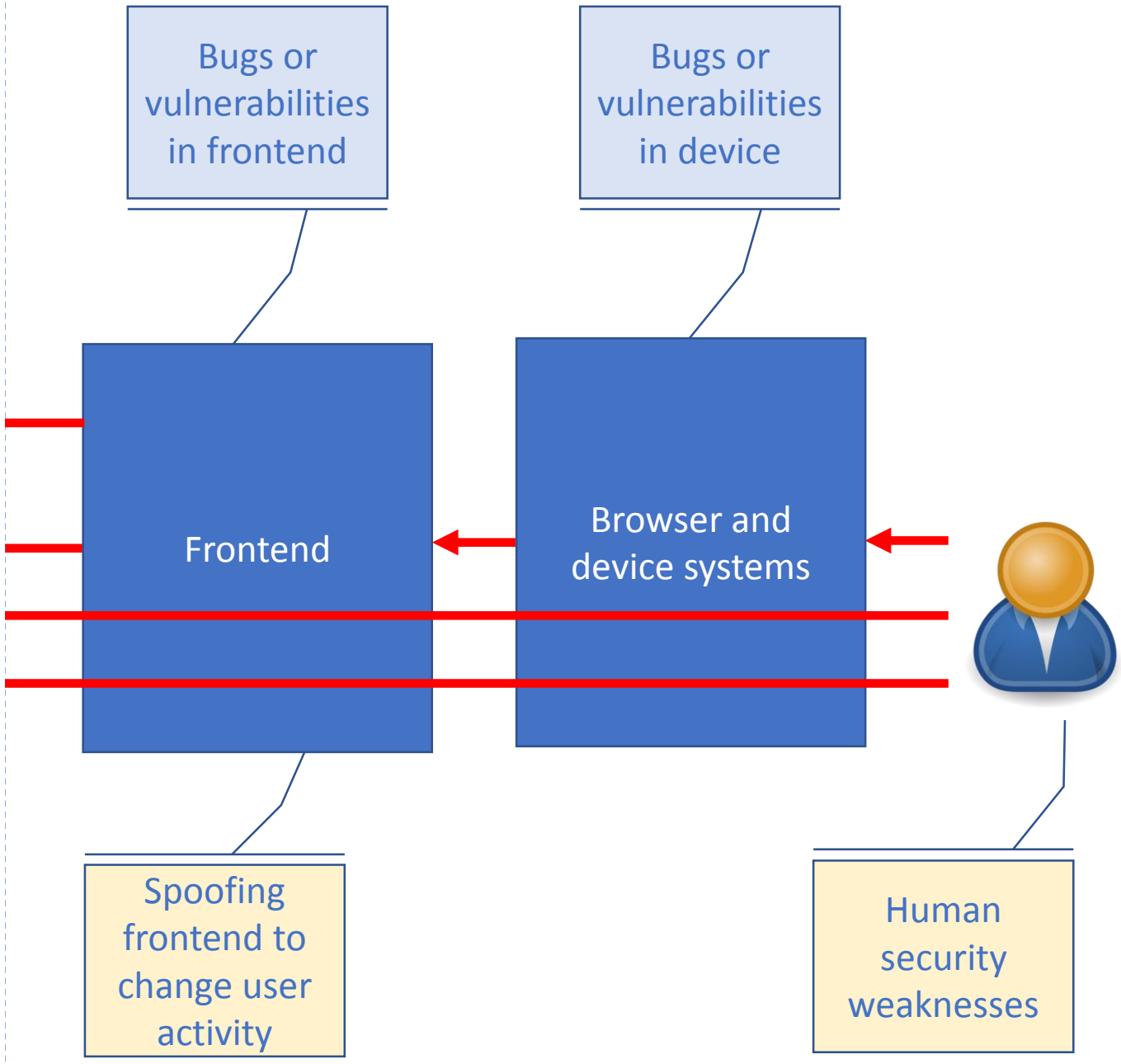

Vulnerabilities in DLT node software
Bugs or vulnerabilities in smart contracts
Bugs or vulnerabilities in frontend
Bugs or vulnerabilities in device
DLT code handling native transactions and smart contract execution
Smart contracts
Frontend
Browser and device systems
Consensus algorithm
Consensus algorithm vulnerabilities
Spoofing frontend to change user activity
Human security weaknesses


Vulnerabilities in DLT node software
Bugs or vulnerabilities in smart contracts
Bugs or vulnerabilities in frontend
Bugs or vulnerabilities in device
DLT code handling native transactions and smart contract execution
Smart contracts
Frontend
Browser and device systems
Consensus algorithm
Consensus algorithm vulnerabilities
Spoofing frontend to change user activity
Human security weaknesses

Traditional Vulnerabilities

1. Human weaknesses

- Passwords/keys ‘on a post-it note’
- Phishing
- Social engineering
- Rubber-hose cryptanalysis

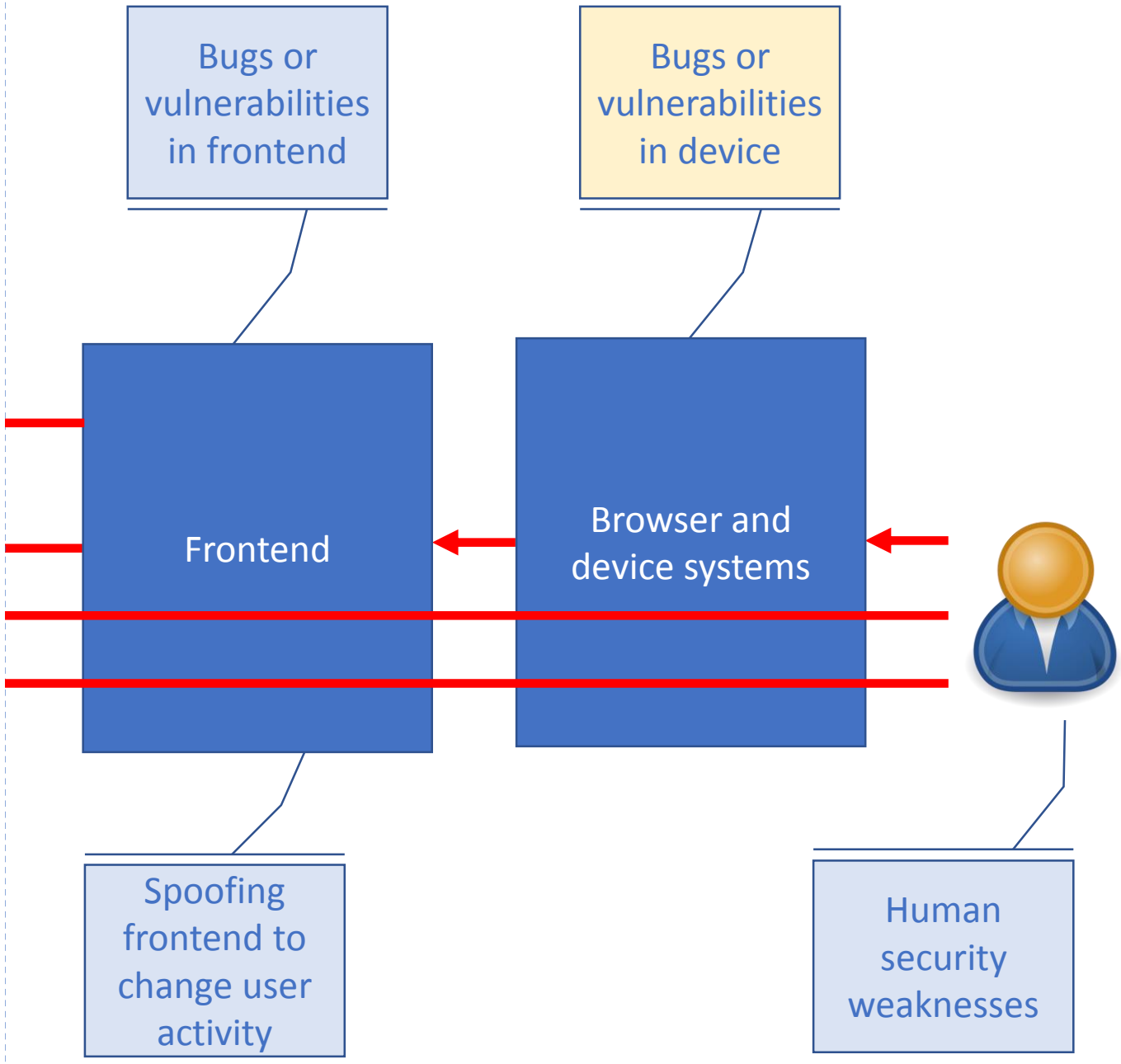


Responsibility: Users.

Traditional Vulnerabilities

2. Platform vulnerabilities

- Due to bugs on the device/browser/operating system/platform.
- Regular updates required to be carried out



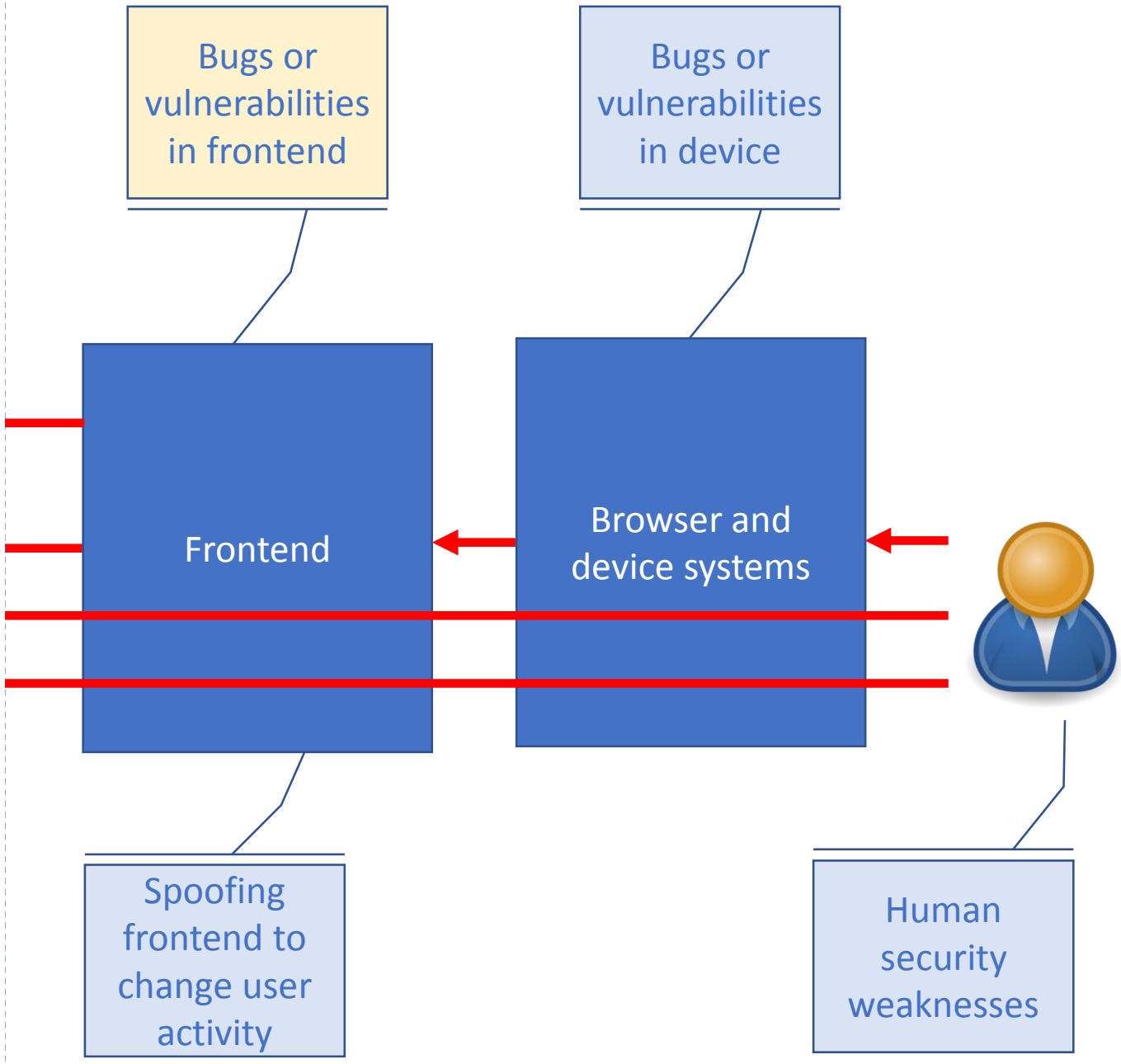
Responsibility: Users.

Traditional Vulnerabilities

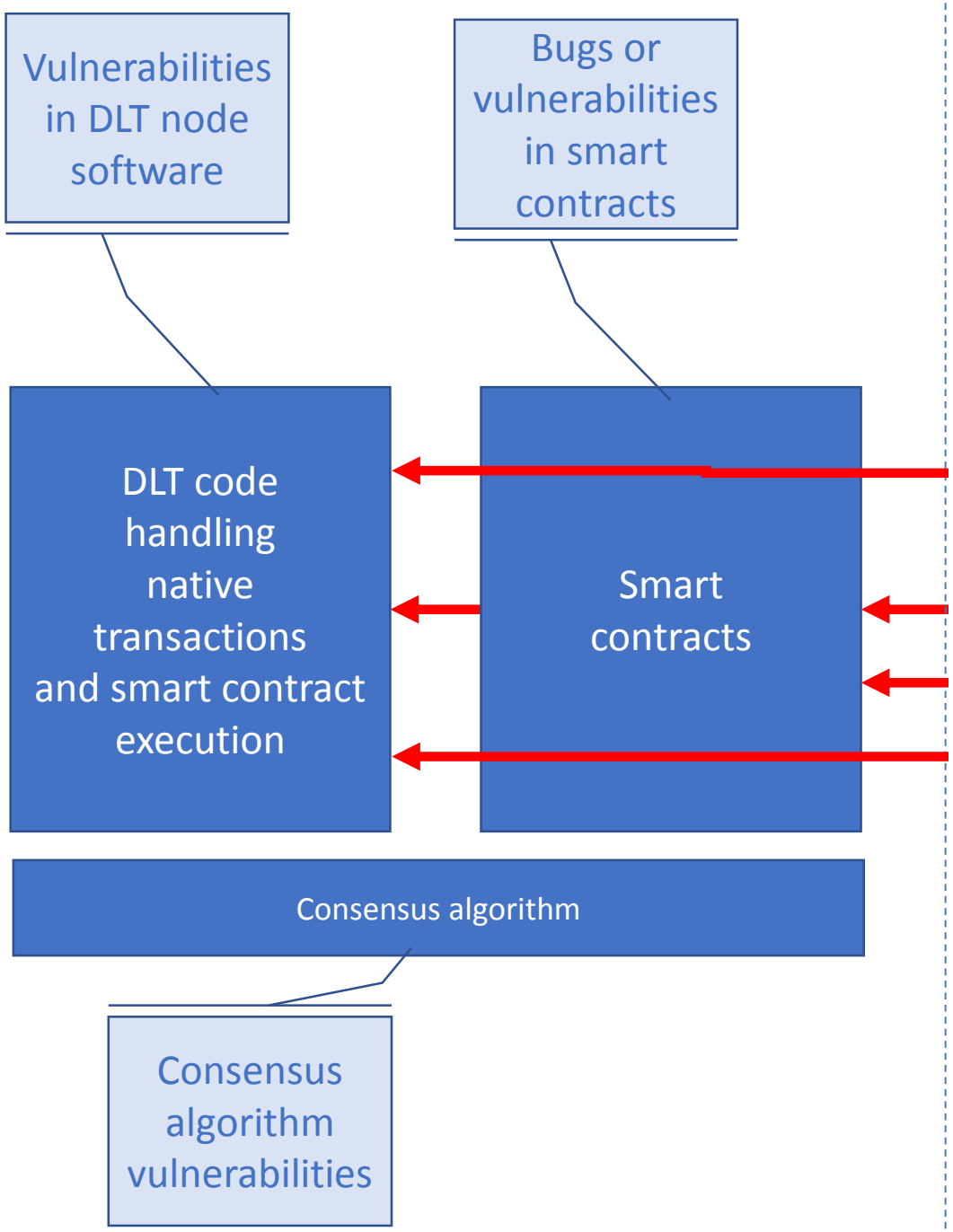
3. Front-end vulnerabilities

- The responsibility of the developers (of the front end), not the users.
- Traditional security measures.
- Handled through code analysis, penetration testing, etc.

Responsibility: Developers of front-end.

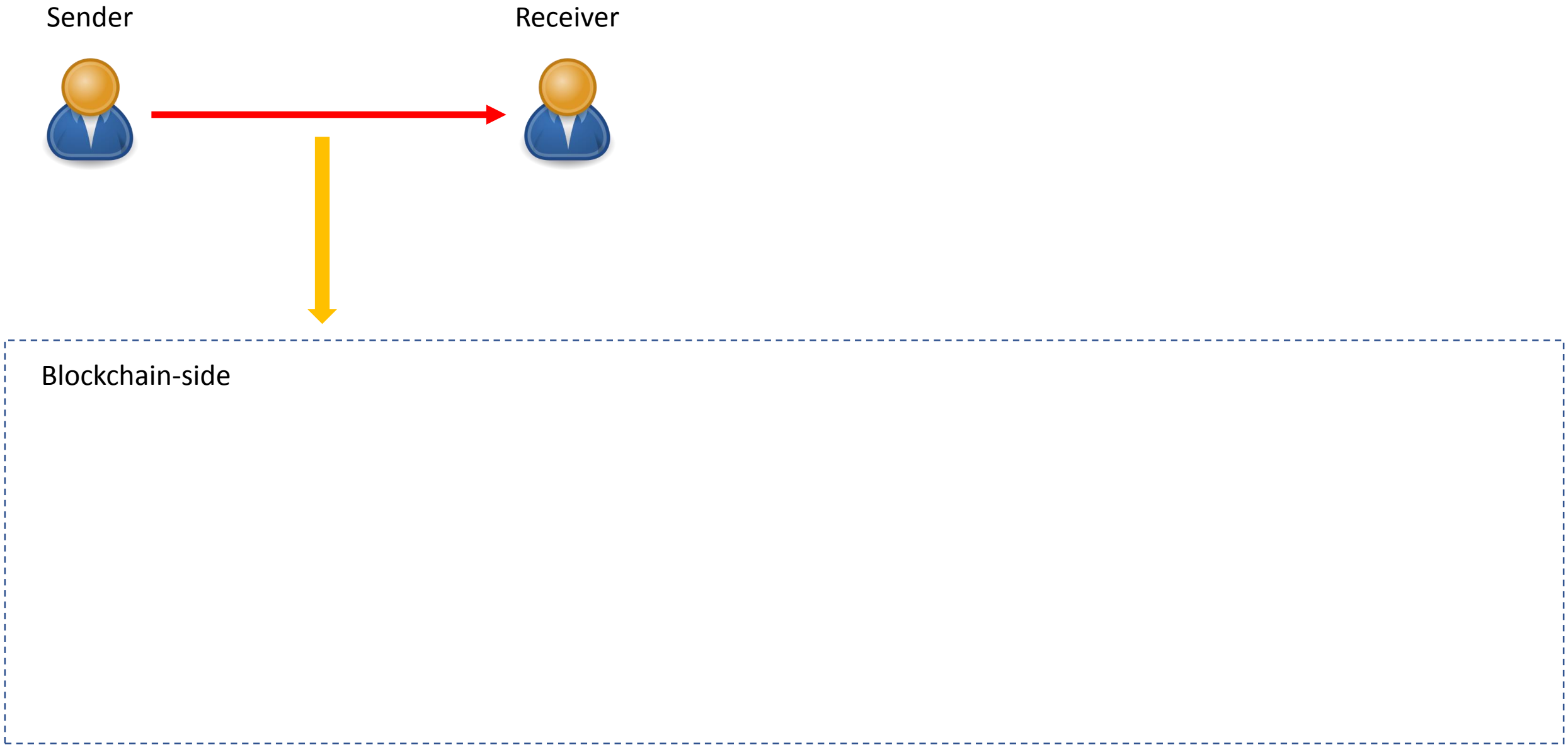


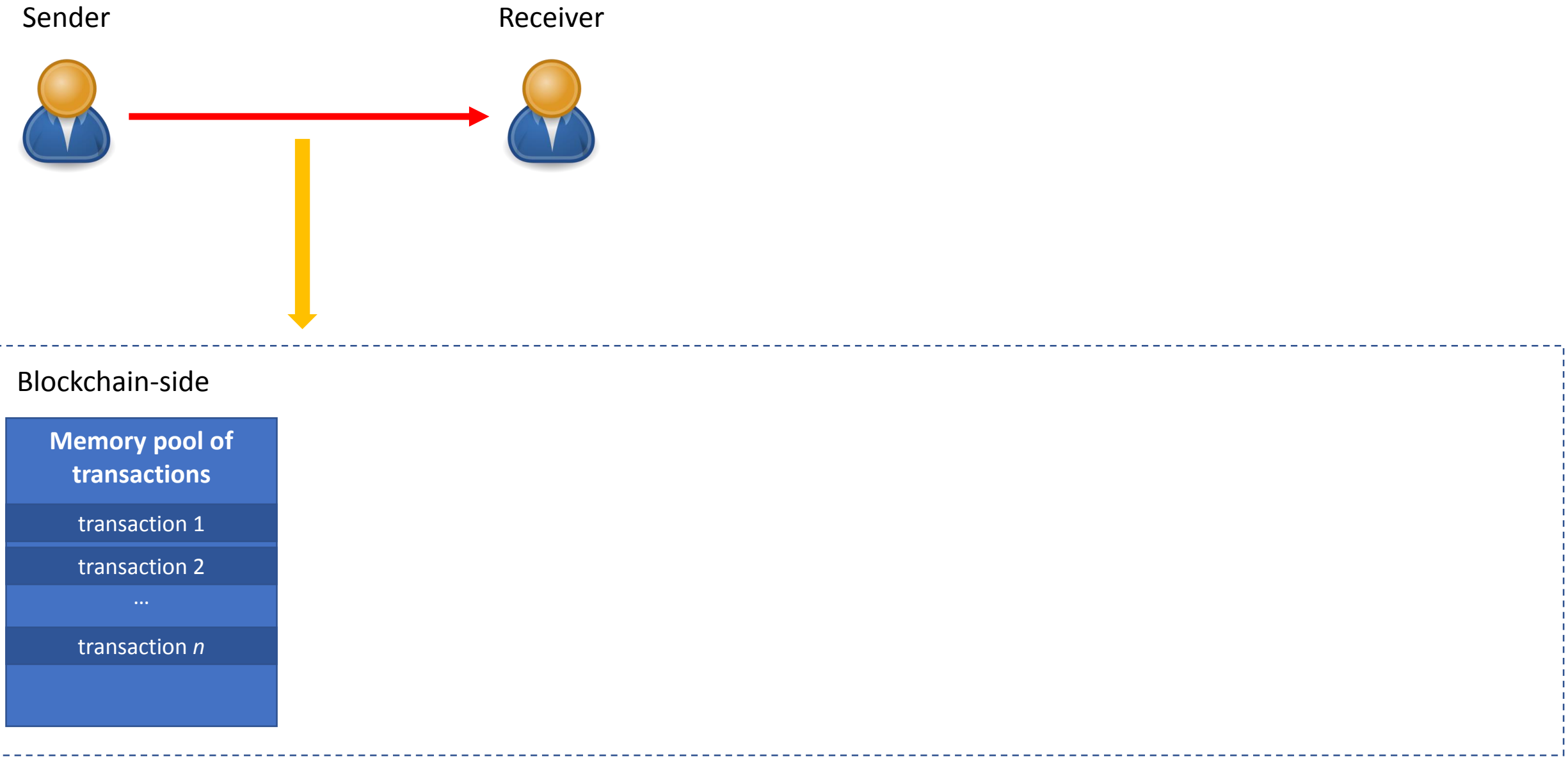
Blockchain-Side Vulnerabilities

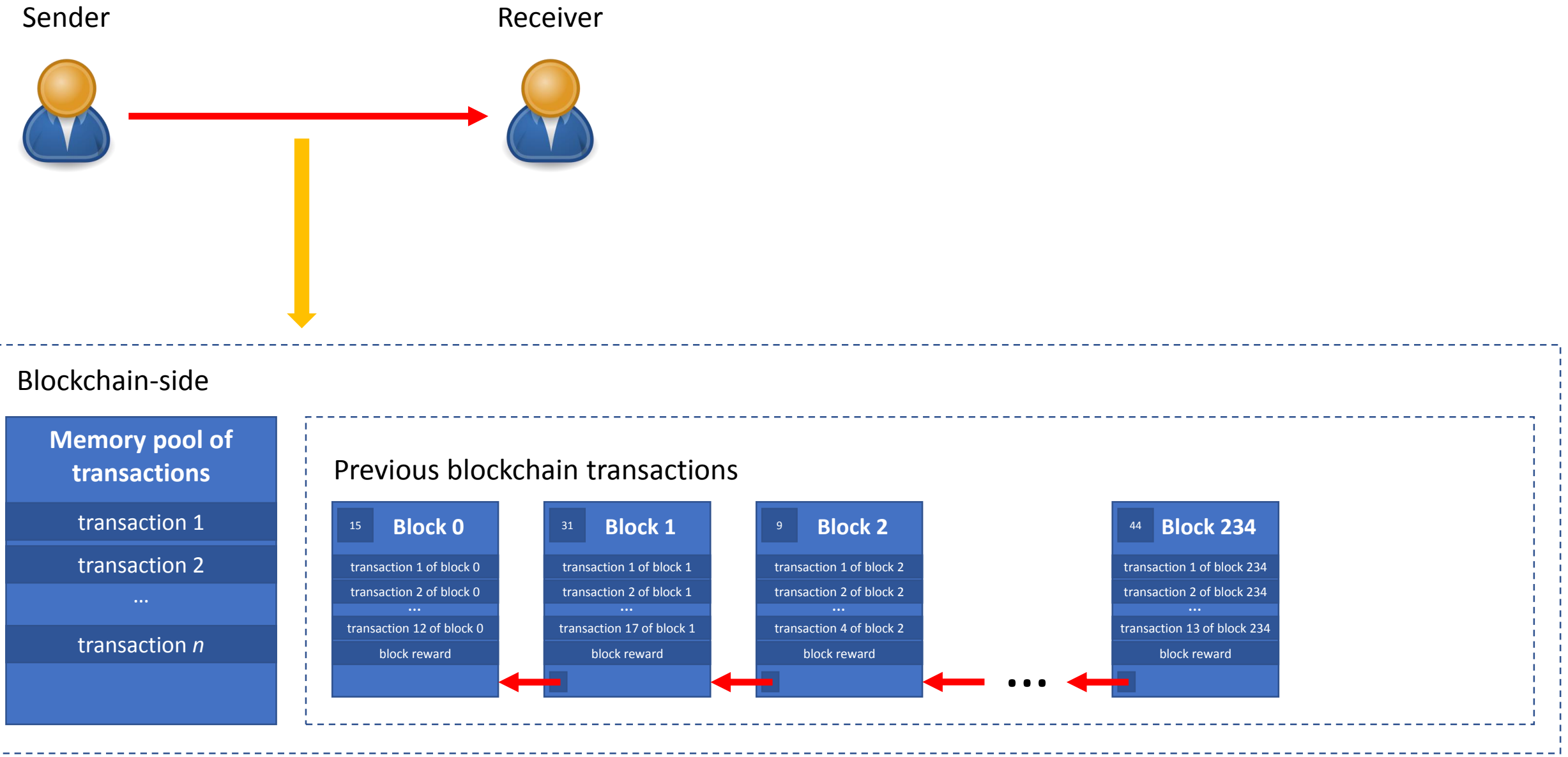


Sender

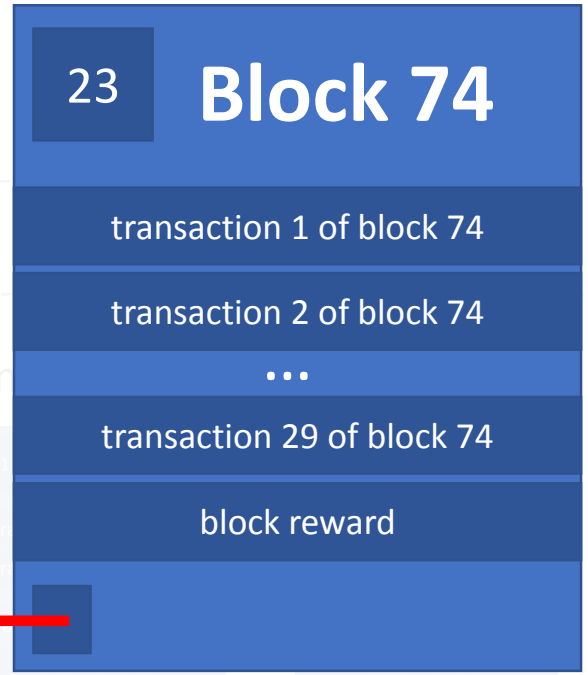
Receiver


Sender
Receiver
Blockchain-side

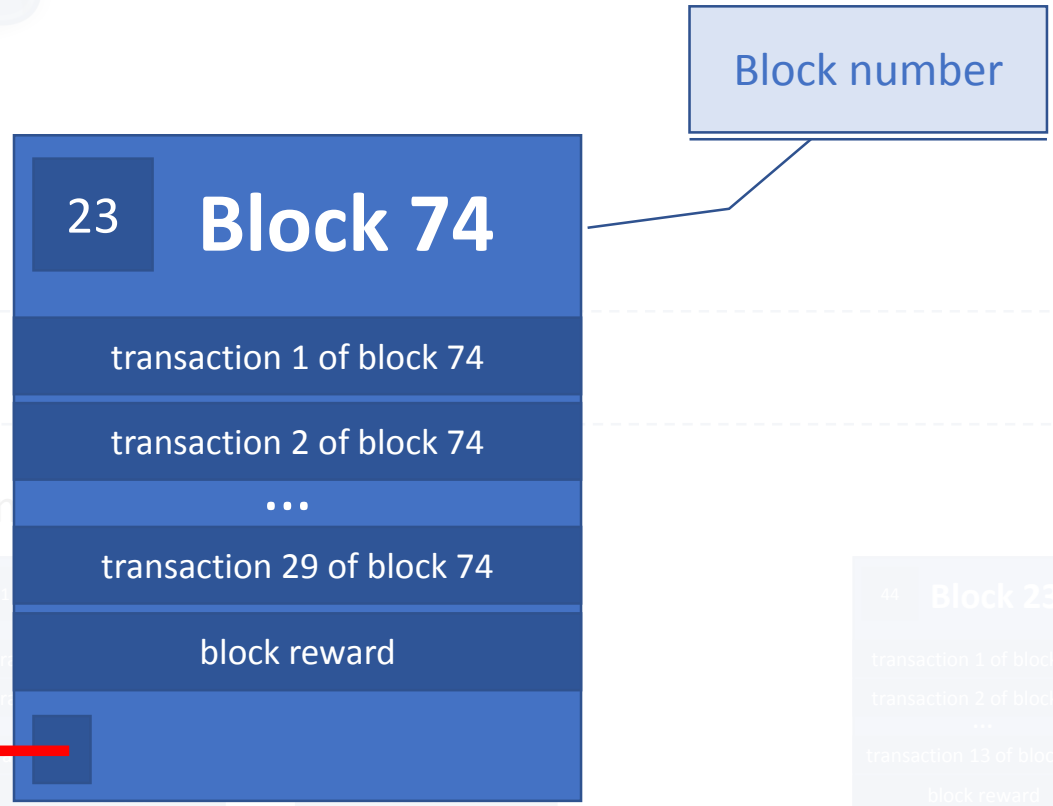

Sender
Receiver
Blockchain-side
Memory pool of transactions
transaction 1
transaction 2
...
transaction n


Sender
Receiver
Blockchain-side
Memory pool of transactions
transaction 1
transaction 2
...
transaction n
Previous blockchain transactions
15
Block 0
transaction 1 of block 0
transaction 2 of block 0
...
transaction 12 of block 0
block reward
31
Block 1
transaction 1 of block 1
transaction 2 of block 1
...
transaction 17 of block 1
block reward
9
Block 2
transaction 1 of block 2
transaction 2 of block 2
...
transaction 4 of block 2
block reward
...
44
Block 234
transaction 1 of block 234
transaction 2 of block 234
...
transaction 13 of block 234
block reward

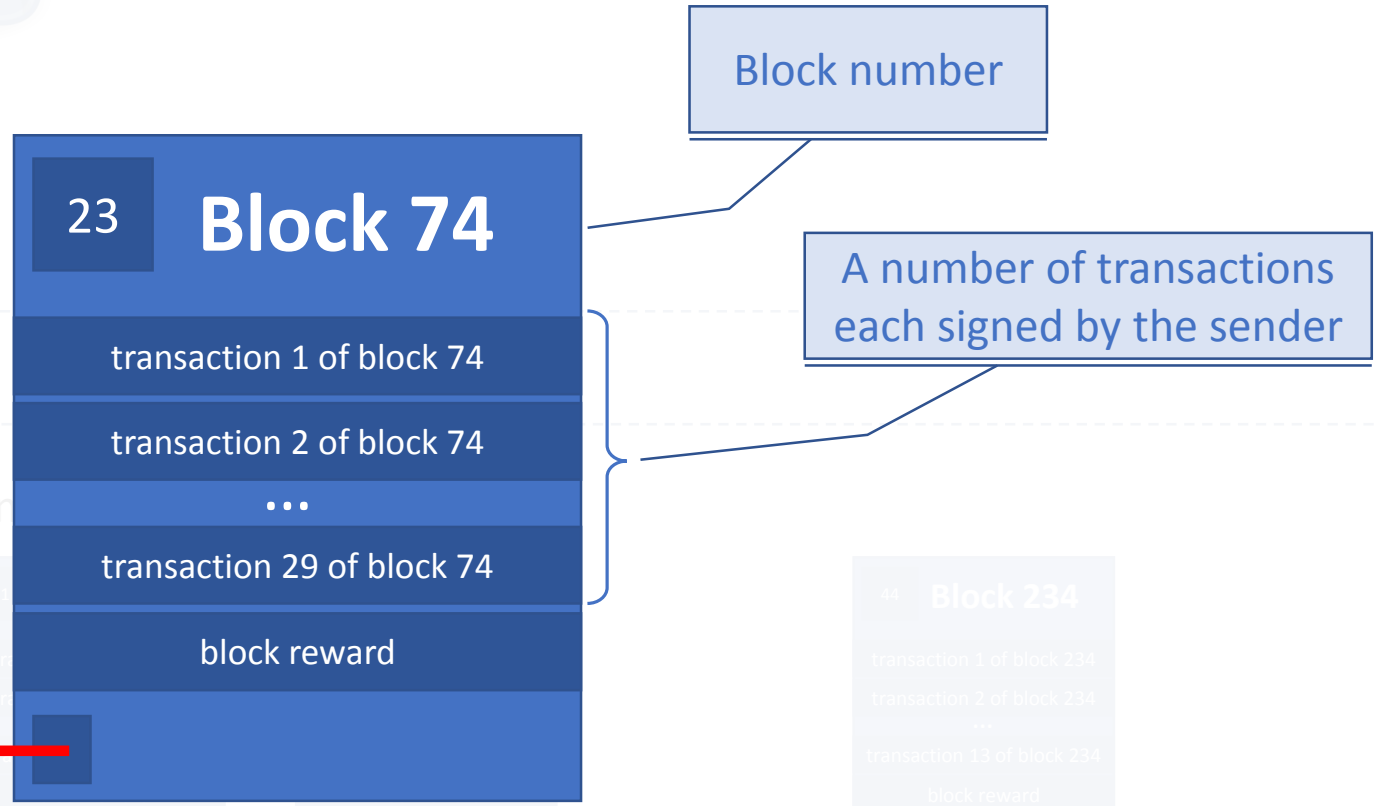
Anatomy of a block



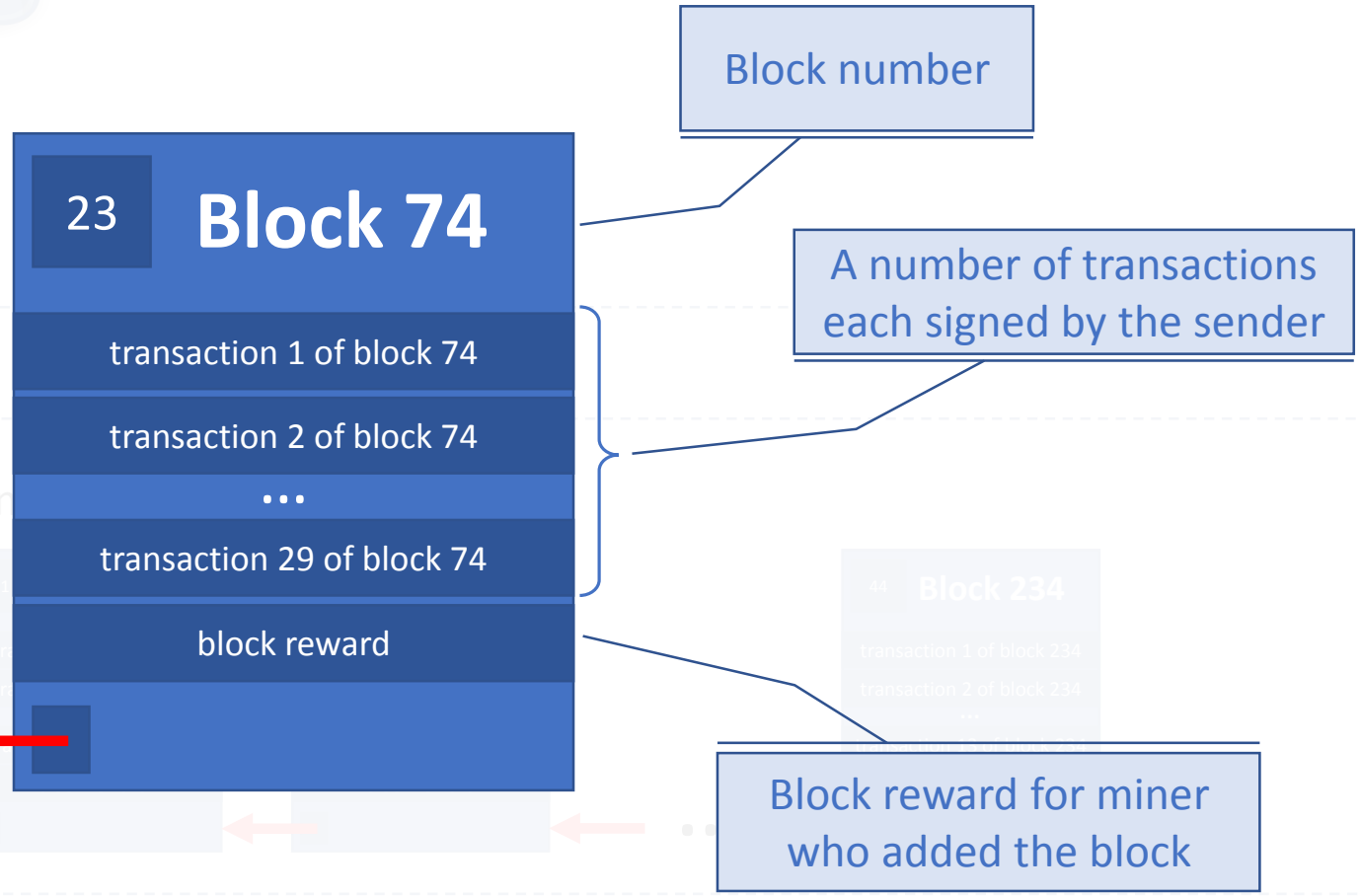
Anatomy of a block



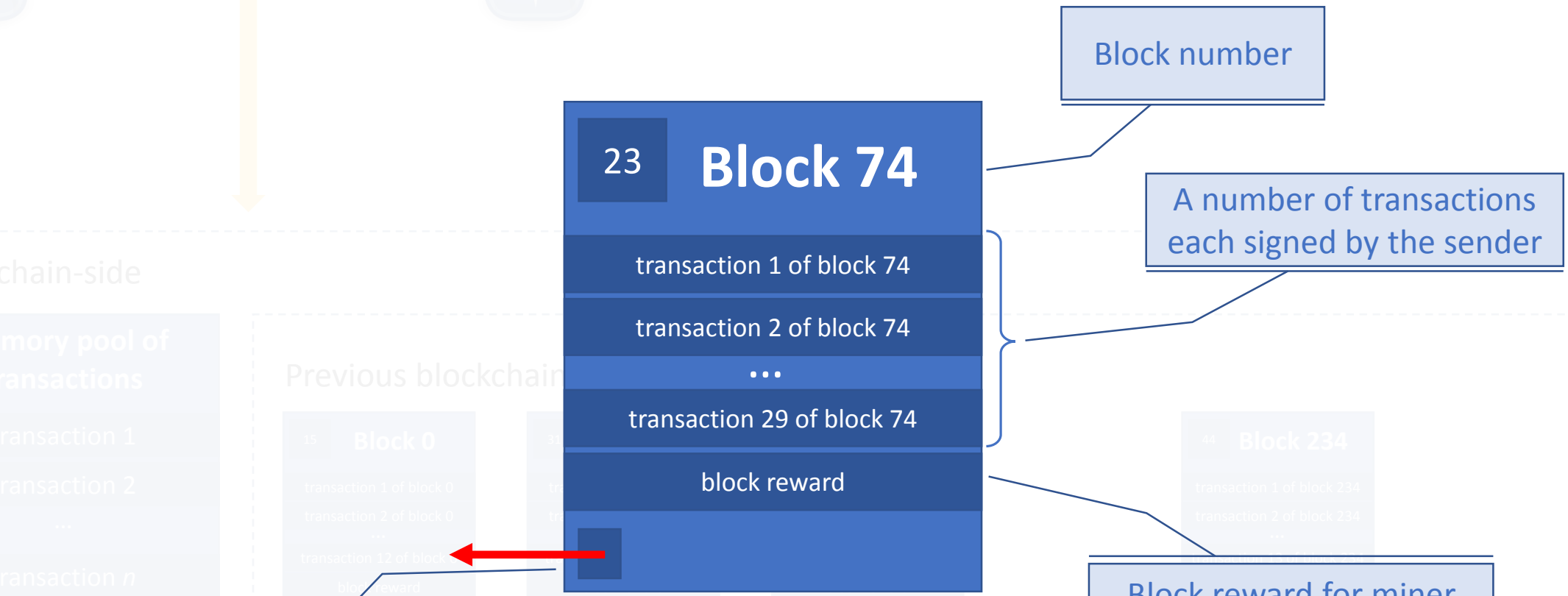
Anatomy of a block



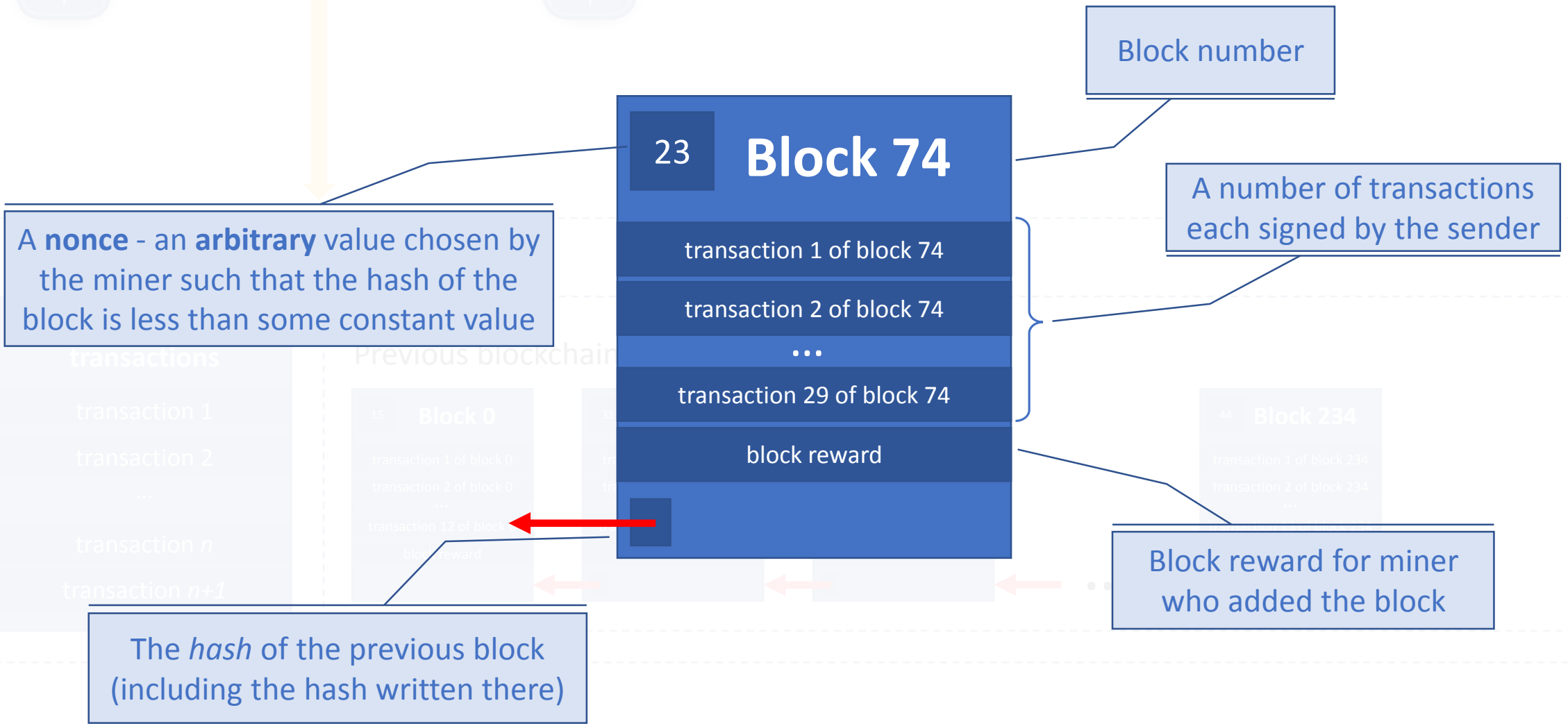
Anatomy of a block

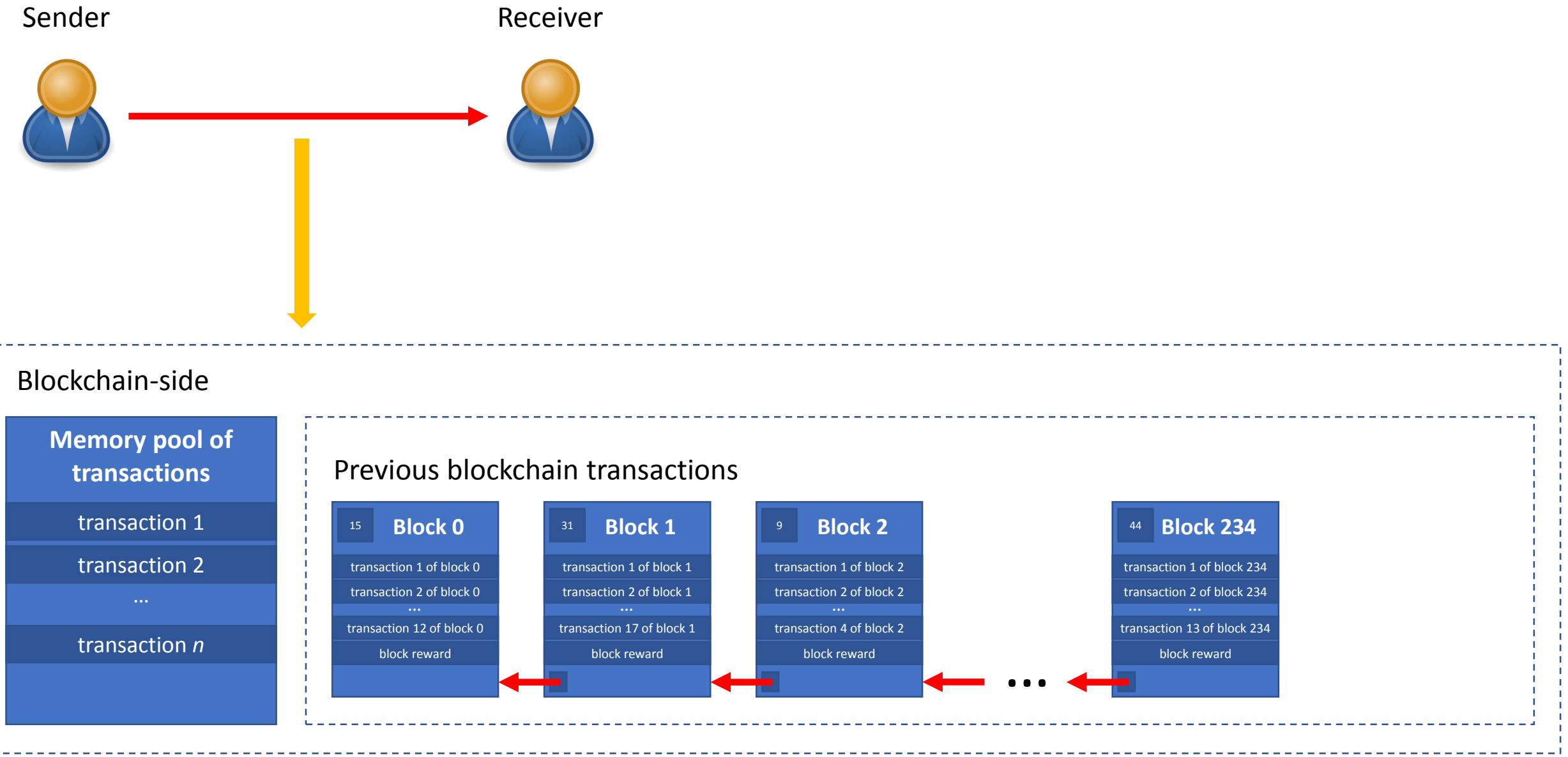


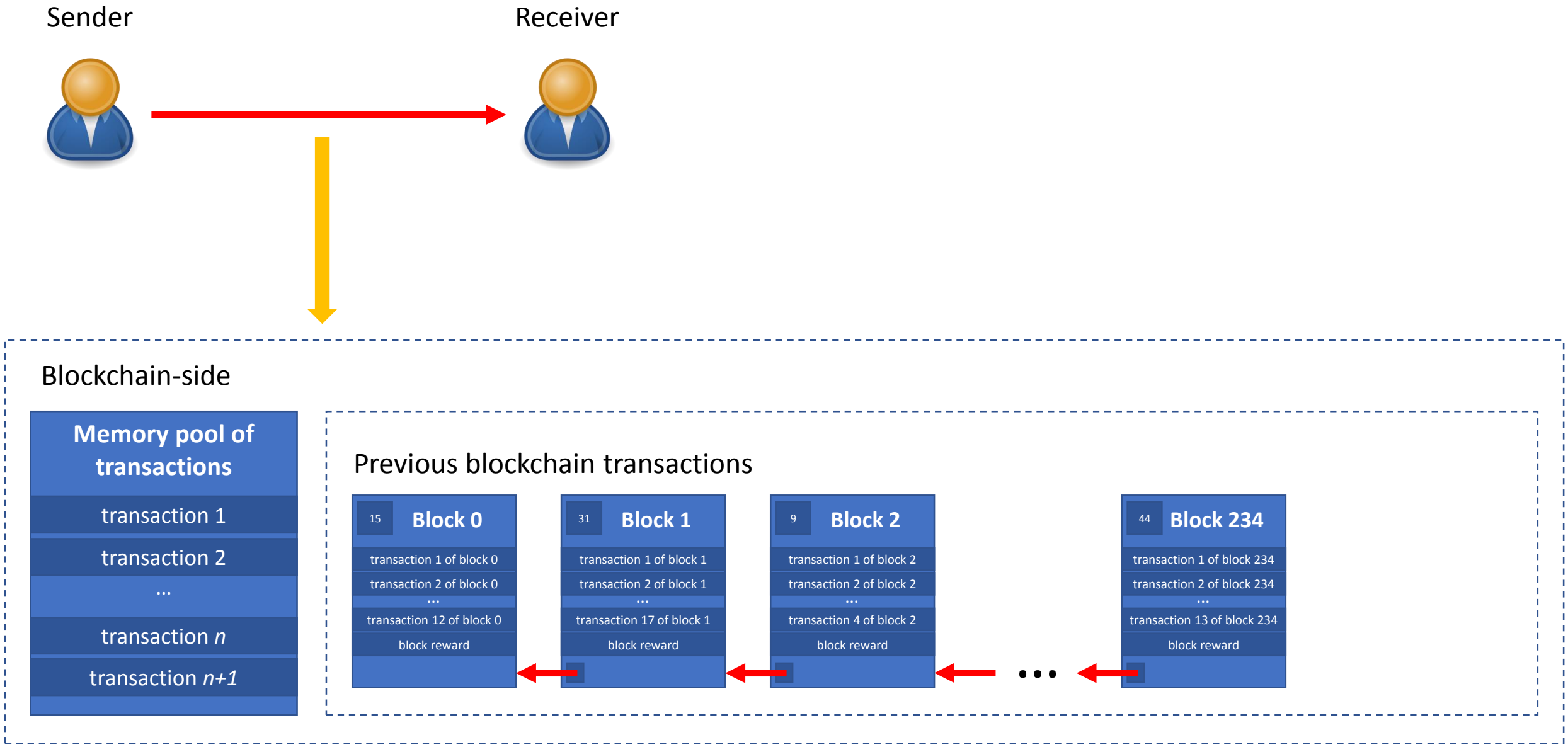
Anatomy of a block



Anatomy of a block




Sender
Receiver
Blockchain-side
Memory pool of transactions
transaction 1
transaction 2
...
transaction n
Previous blockchain transactions
15
Block 0
transaction 1 of block 0
transaction 2 of block 0
...
transaction 12 of block 0
block reward
31
Block 1
transaction 1 of block 1
transaction 2 of block 1
...
transaction 17 of block 1
block reward
9
Block 2
transaction 1 of block 2
transaction 2 of block 2
...
transaction 4 of block 2
block reward
...
44
Block 234
transaction 1 of block 234
transaction 2 of block 234
...
transaction 13 of block 234
block reward


Sender
Receiver
Blockchain-side
Memory pool of transactions
transaction 1
transaction 2
...
transaction n
transaction n+1
Previous blockchain transactions
15
Block 0
transaction 1 of block 0
transaction 2 of block 0
...
transaction 12 of block 0
block reward
31
Block 1
transaction 1 of block 1
transaction 2 of block 1
...
transaction 17 of block 1
block reward
9
Block 2
transaction 1 of block 2
transaction 2 of block 2
...
transaction 4 of block 2
block reward
...
44
Block 234
transaction 1 of block 234
transaction 2 of block 234
...
transaction 13 of block 234
block reward

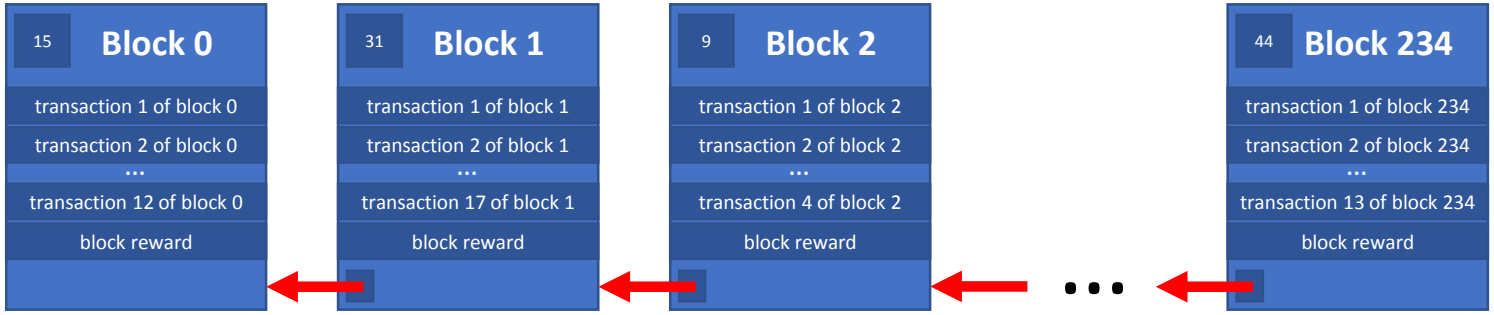
Miner

Blockchain-side

Memory pool of transactions

- transaction 1
- transaction 2
- ...
- transaction *n*
- transaction *n+1*

Previous blockchain transactions



Miner

1. **Choose some pending transactions.**

Blockchain-side

Memory pool of transactions

- transaction 1
- transaction 2
- ...
- transaction *n*
- transaction *n+1*

Previous blockchain transactions

15 Block 0	31 Block 1	9 Block 2	44 Block 234
transaction 1 of block 0	transaction 1 of block 1	transaction 1 of block 2	transaction 1 of block 234
transaction 2 of block 0	transaction 2 of block 1	transaction 2 of block 2	transaction 2 of block 234
...	...	...	...
transaction 12 of block 0	transaction 17 of block 1	transaction 4 of block 2	transaction 13 of block 234
block reward	block reward	block reward	block reward

Miner

transaction 2

transaction 5

...

transaction *29*

transaction *n+1*

1. **Choose some pending transactions.**

Blockchain-side

Memory pool of transactions

transaction 1

transaction 2

...

transaction *n*

transaction *n+1*

Previous blockchain transactions

15 **Block 0**

transaction 1 of block 0

transaction 2 of block 0

...

transaction 12 of block 0

block reward

31 **Block 1**

transaction 1 of block 1

transaction 2 of block 1

...

transaction 17 of block 1

block reward

9 **Block 2**

transaction 1 of block 2

transaction 2 of block 2

...

transaction 4 of block 2

block reward

...

44 **Block 234**

transaction 1 of block 234

transaction 2 of block 234

...

transaction 13 of block 234

block reward

Miner

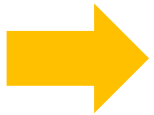
transaction 2

transaction 5

...

transaction *29*

transaction *n+1*



hash of block 234

1. Choose some pending transactions.
2. Put them in a new block.

Blockchain-side

Memory pool of transactions

transaction 1

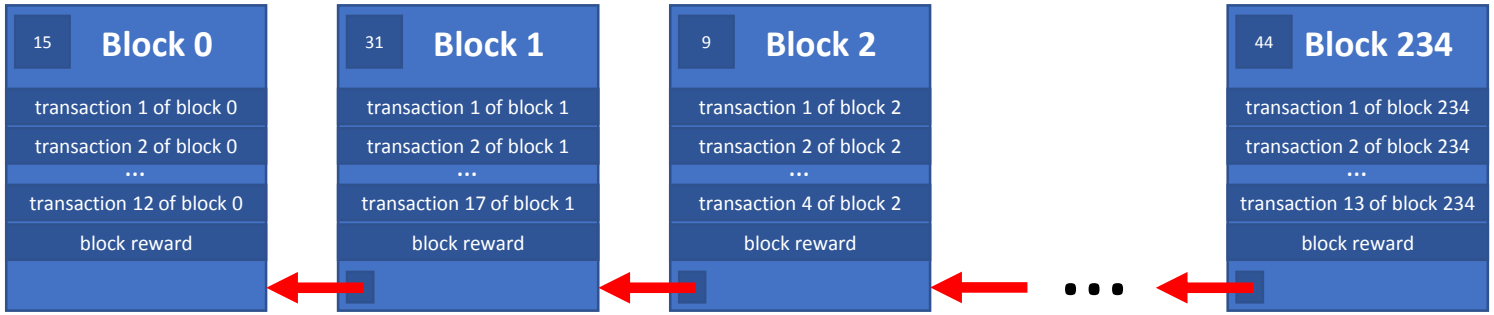
transaction 2

...

transaction *n*

transaction *n+1*

Previous blockchain transactions



Miner

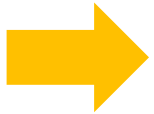
transaction 2

transaction 5

...

transaction *29*

transaction *n+1*



hash of block 234

1. Choose some pending transactions.
2. Put them in a new block.
3. Find a nonce such that hash of the block is less than N.

Blockchain-side

Memory pool of transactions

transaction 1

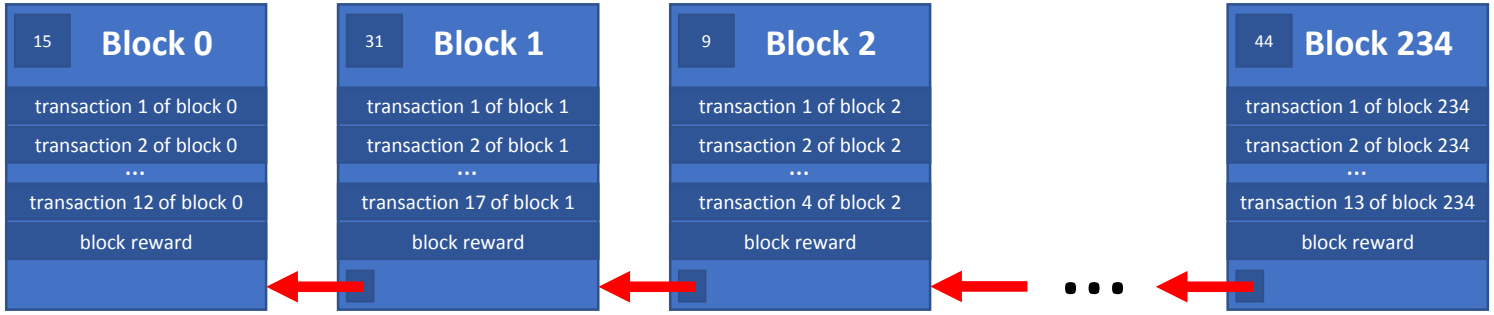
transaction 2

...

transaction *n*

transaction *n+1*

Previous blockchain transactions



Miner

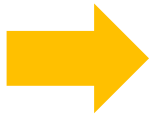
transaction 2

transaction 5

...

transaction *29*

transaction *n+1*



hash of block 234

1. Choose some pending transactions.
2. Put them in a new block.
3. Find a nonce such that hash of the block is less than N.
4. Broadcast the solution.

Blockchain-side

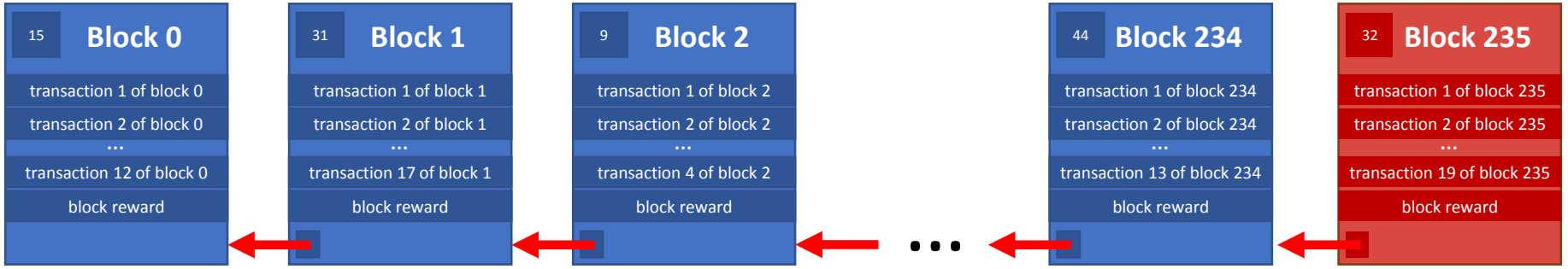
Memory pool of transactions

transaction 1

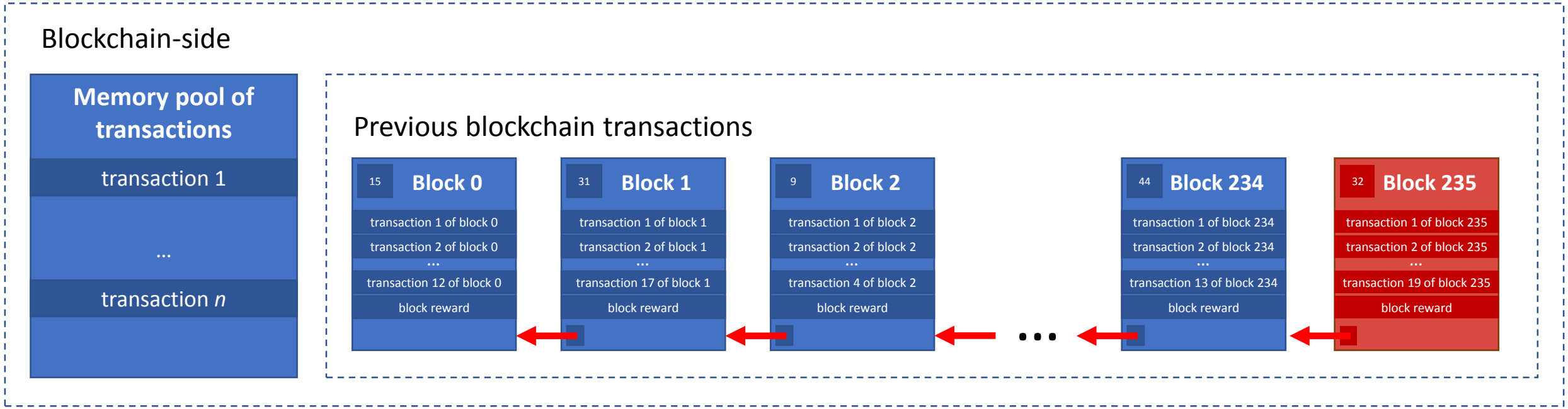
...

transaction *n*

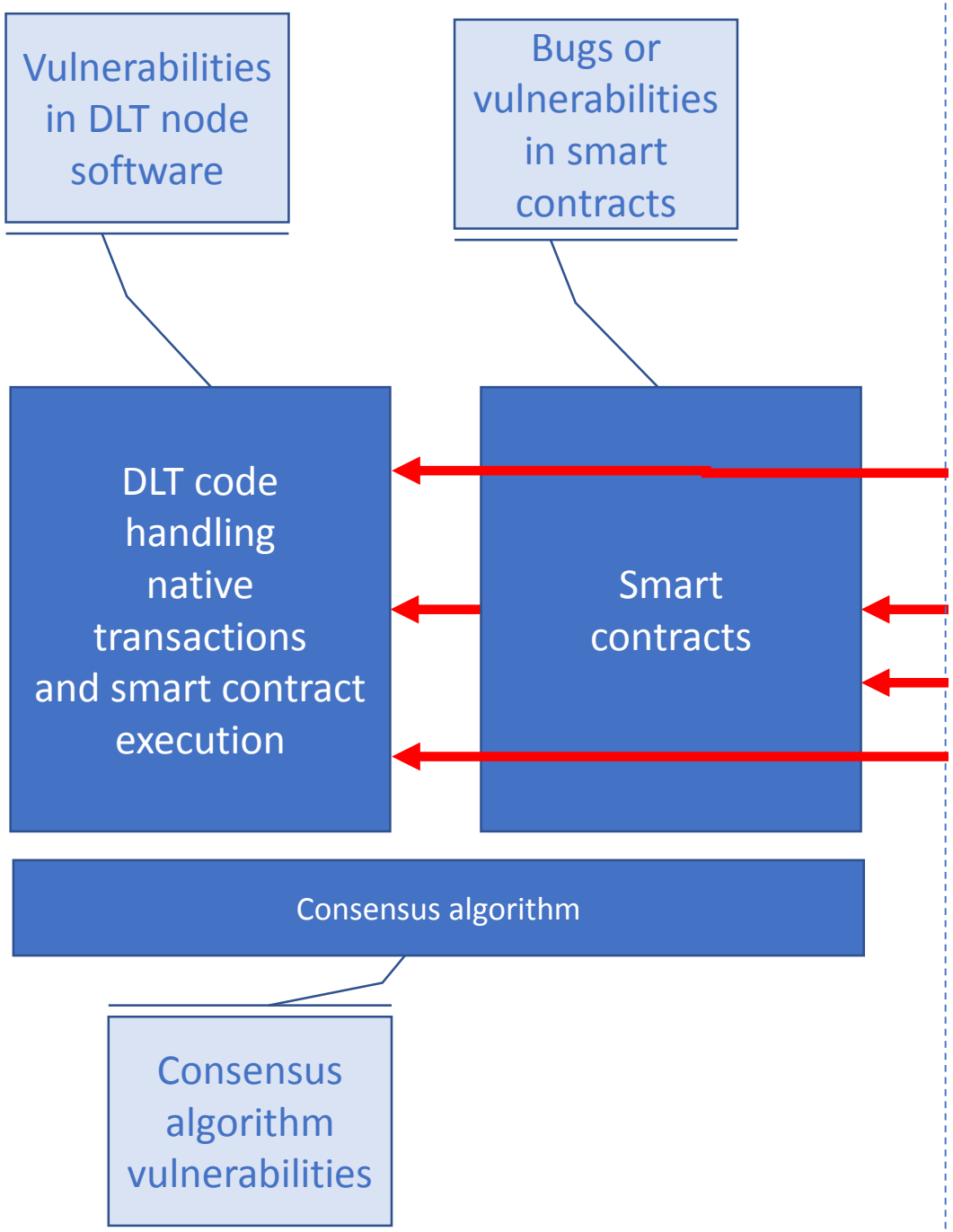
Previous blockchain transactions

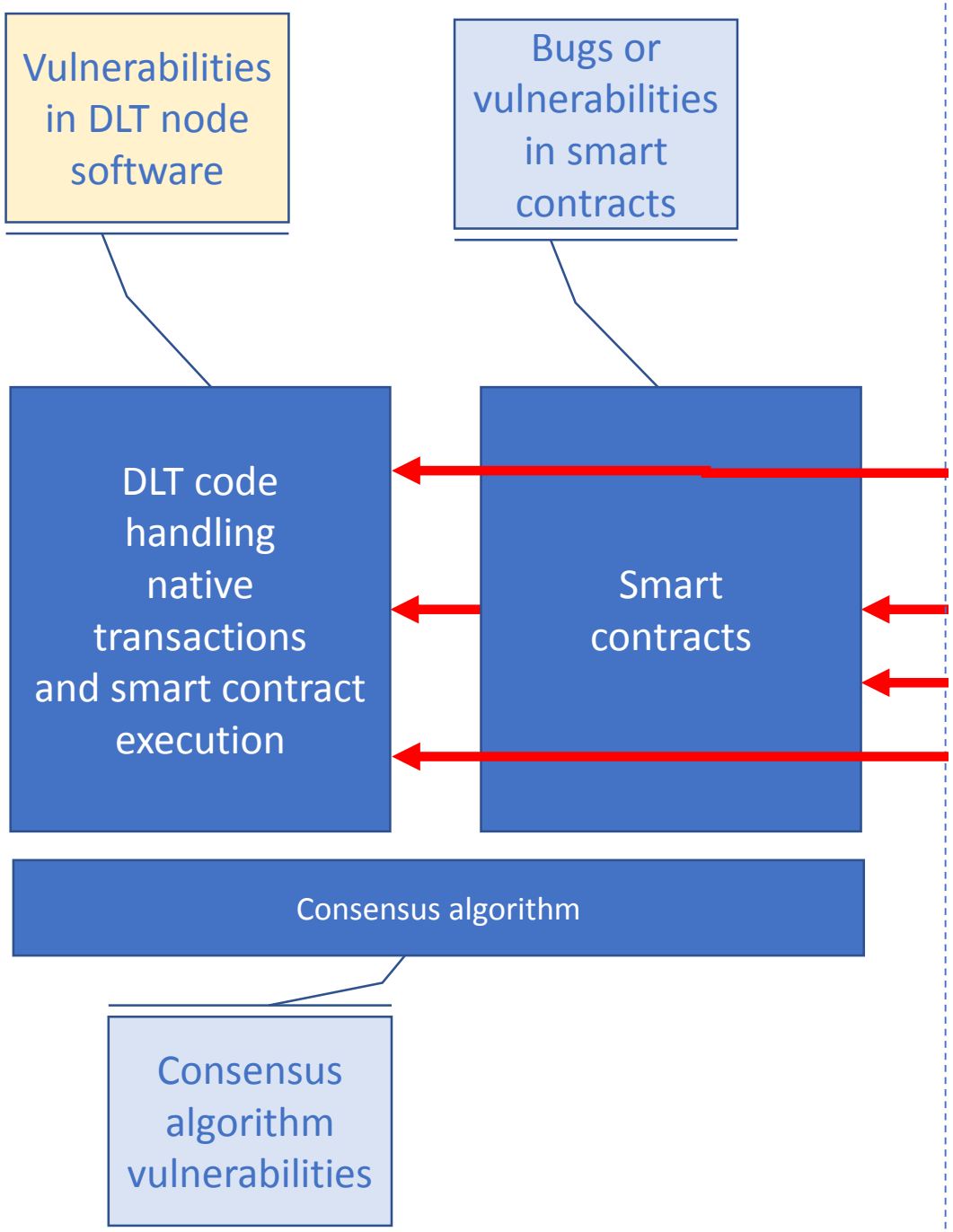


What if two miners find a solution at the same time?
Some nodes will adopt one solution, others the other.
As soon as one is extended further, that is preferred (longest chain wins).



Blockchain-Side Vulnerabilities



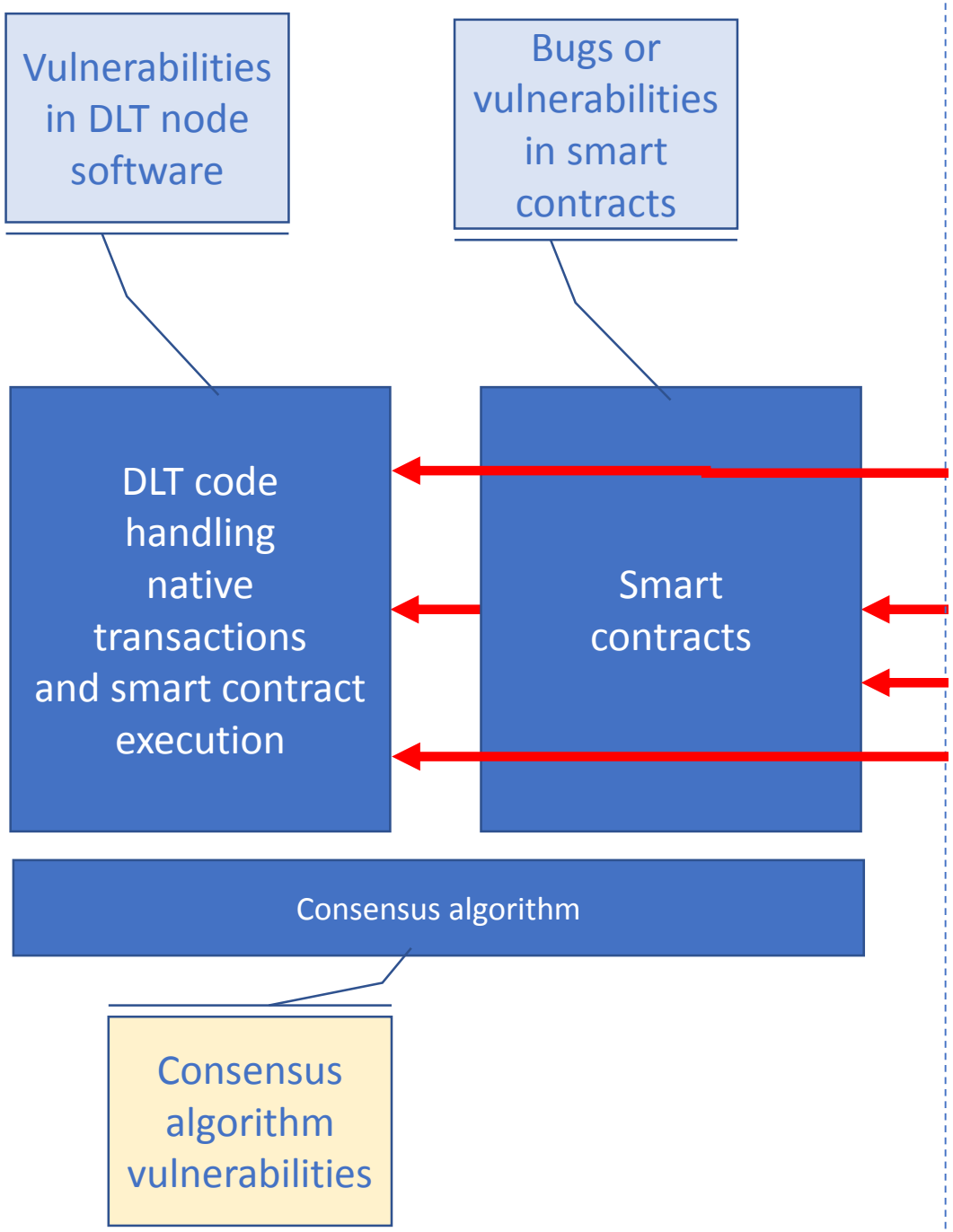


Blockchain-Side Vulnerabilities

4. DLT node vulnerabilities

- DLT node handles communication and protocol to ensure immutability, fairness, etc.
- Such vulnerabilities can also be due to constituent parts e.g. libraries used.
- (Surprisingly) rare instances.

Responsibility: DLT developers.

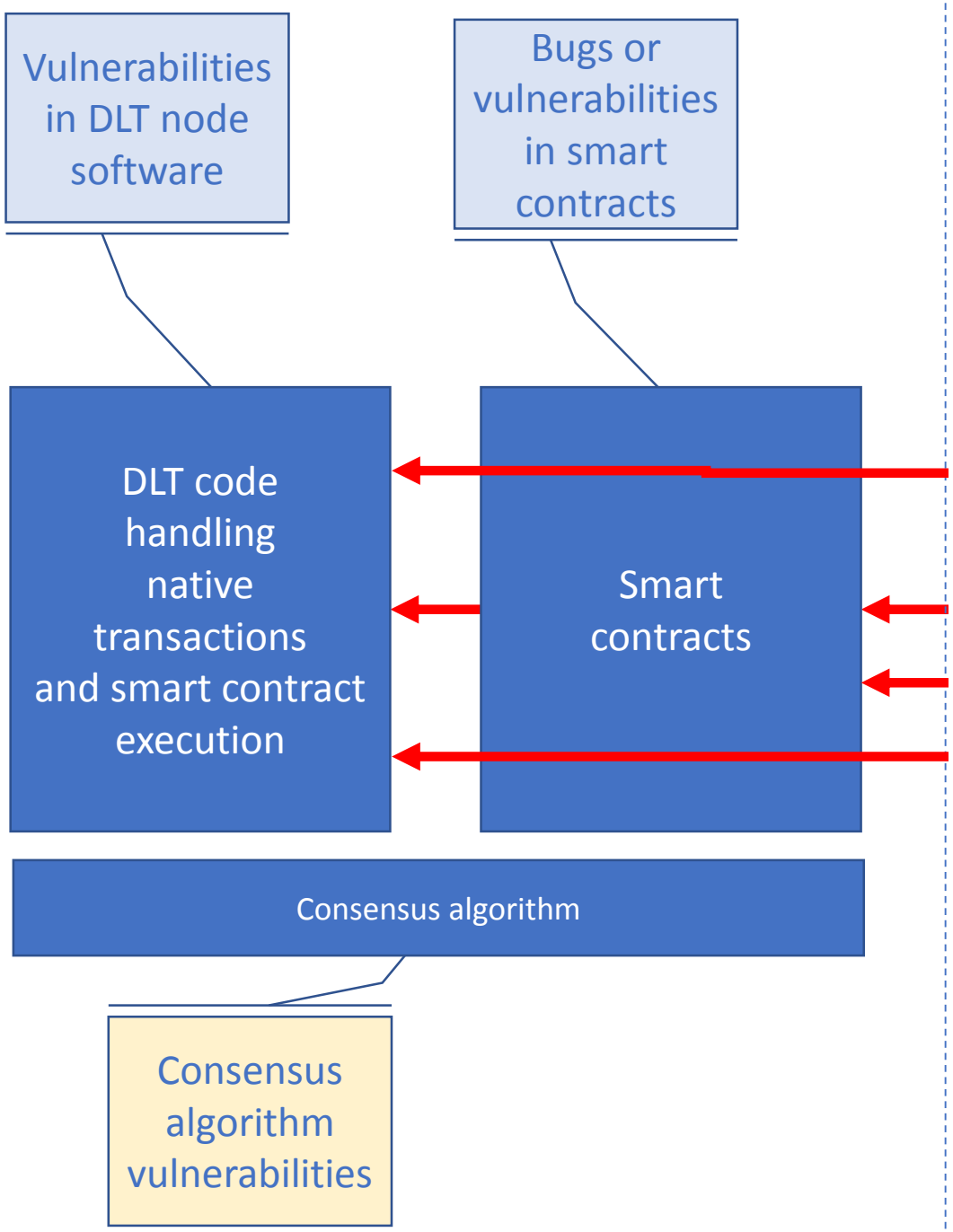


Blockchain-Side Vulnerabilities

5. Consensus algorithm vulnerabilities

- Selfish mining by not sharing found blocks immediately.
- Network attacks aiming at disrupting or manipulating connectivity between nodes at connection time or later on.
- DDOS attacks disrupting miner activity by flooding with micro-transactions or the memory pool of transactions.
- 51% attack aiming at taking over the network.

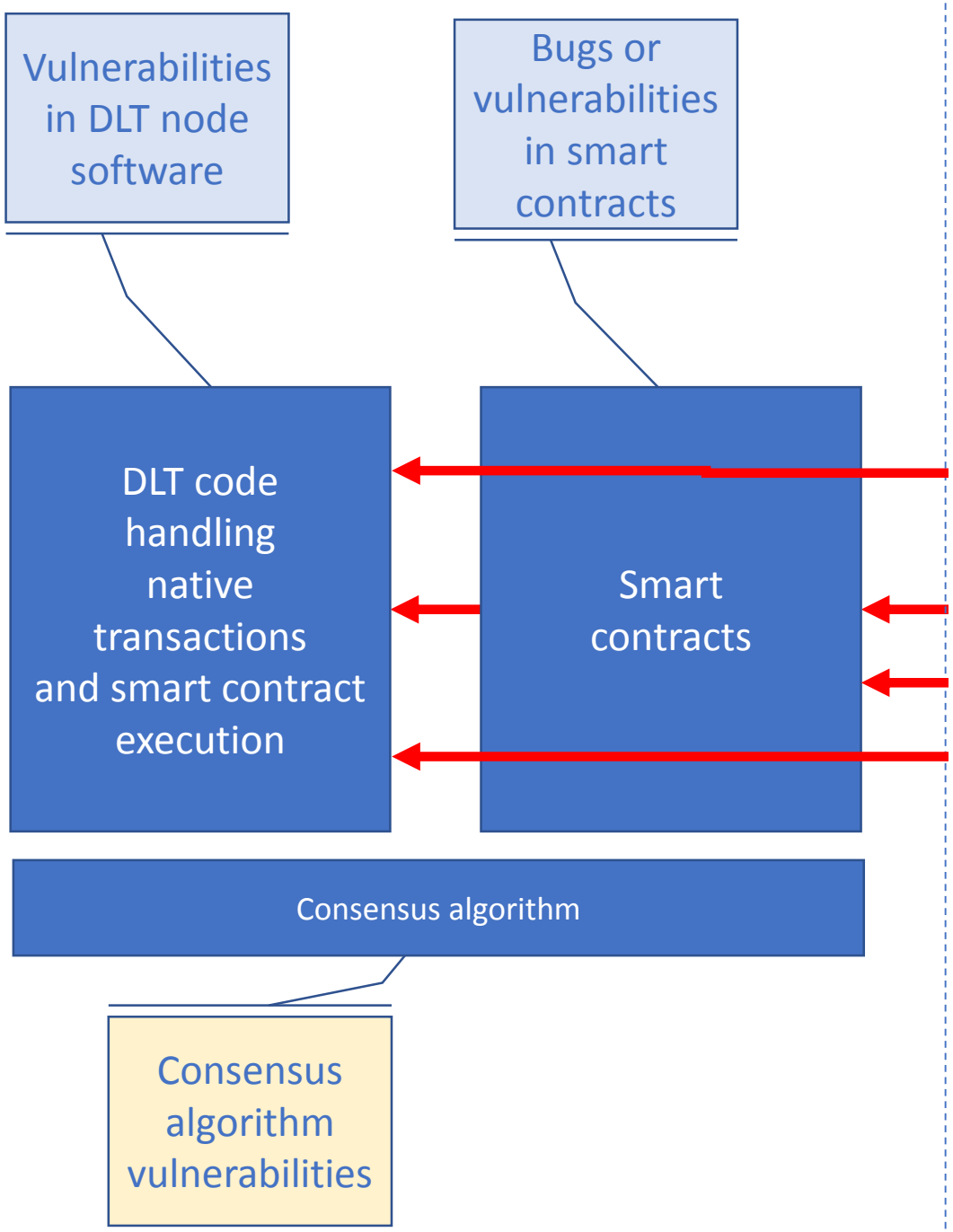
Responsibility: Consensus protocol designers.



Blockchain-Side Vulnerabilities

51% attacks

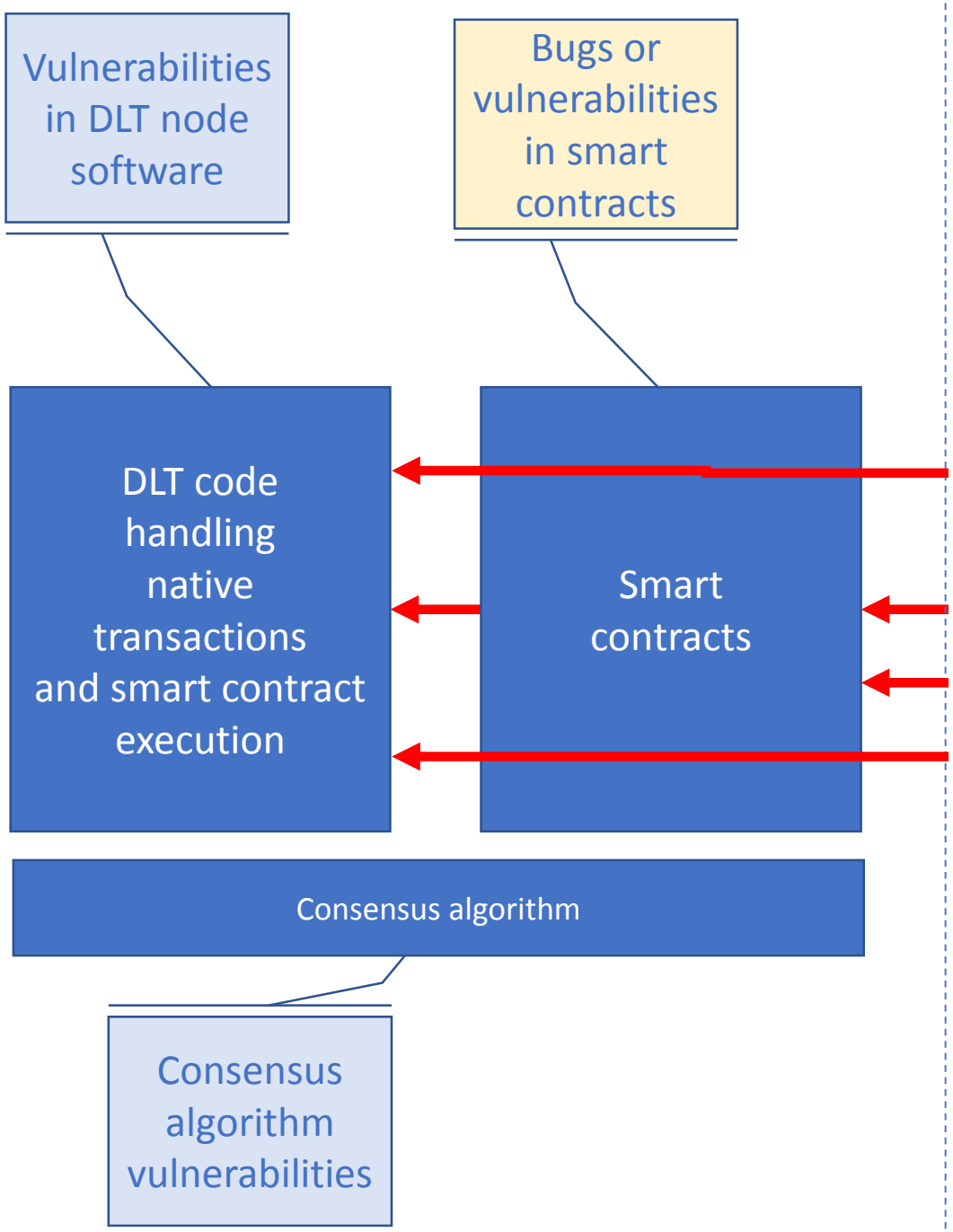
- Power to prevent transactions from taking place.
- Power to 'reverse' transactions – double spending.
- Power to fork the network.
- Power to prevent others from finding blocks.



Blockchain-Side Vulnerabilities

51% attacks

- It's even worse than it sounds:
  - Owning 51% for a short period of time suffices for many attacks.
  - Hash power can be rented.
  - Some attacks have high chance of success even with less than 50% of network computational power.

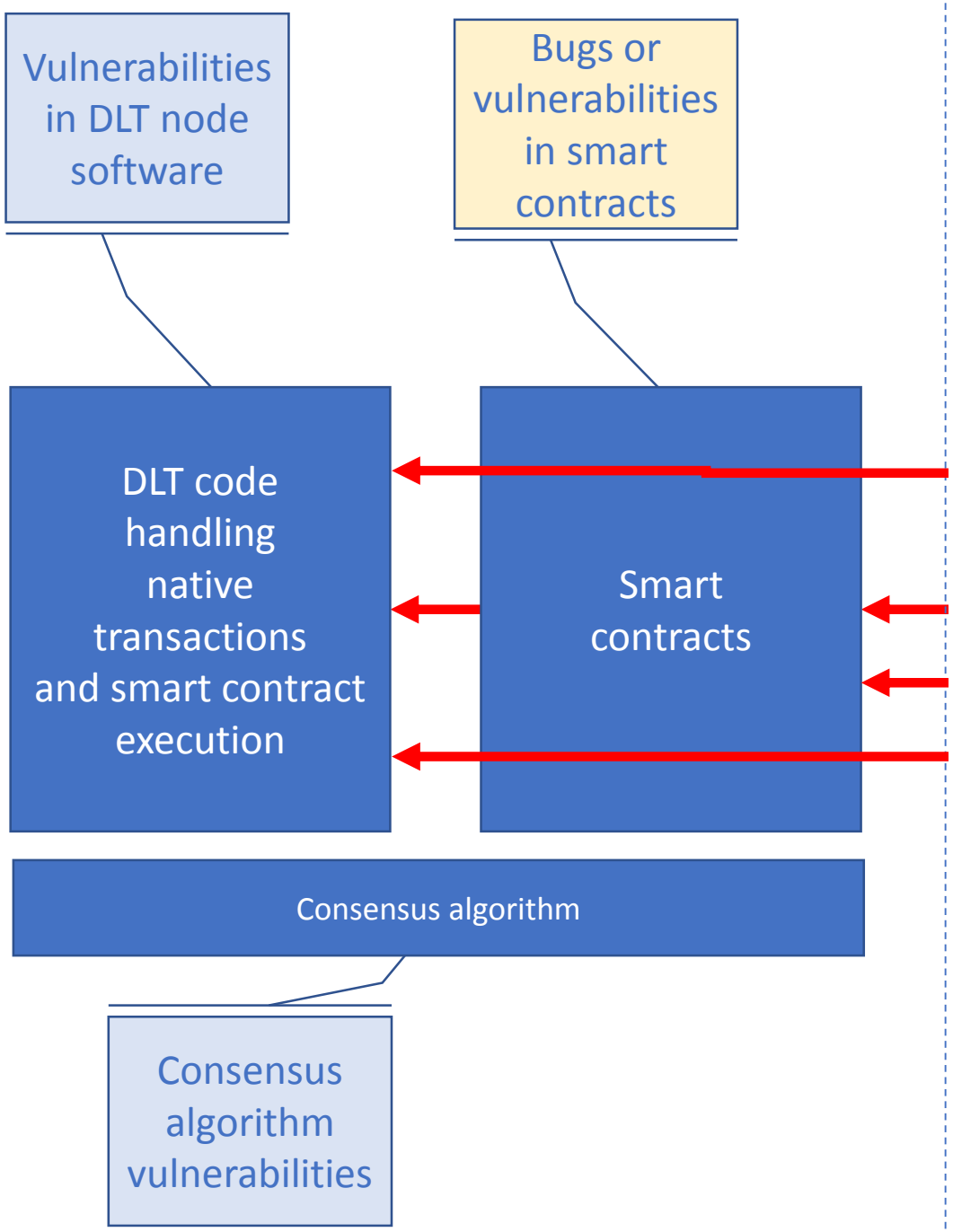


Blockchain-Side Vulnerabilities

6. Smart contract vulnerabilities

- Bugs and deviations from intended functionality.
- Platform or programming language vulnerabilities.
- Malicious code.

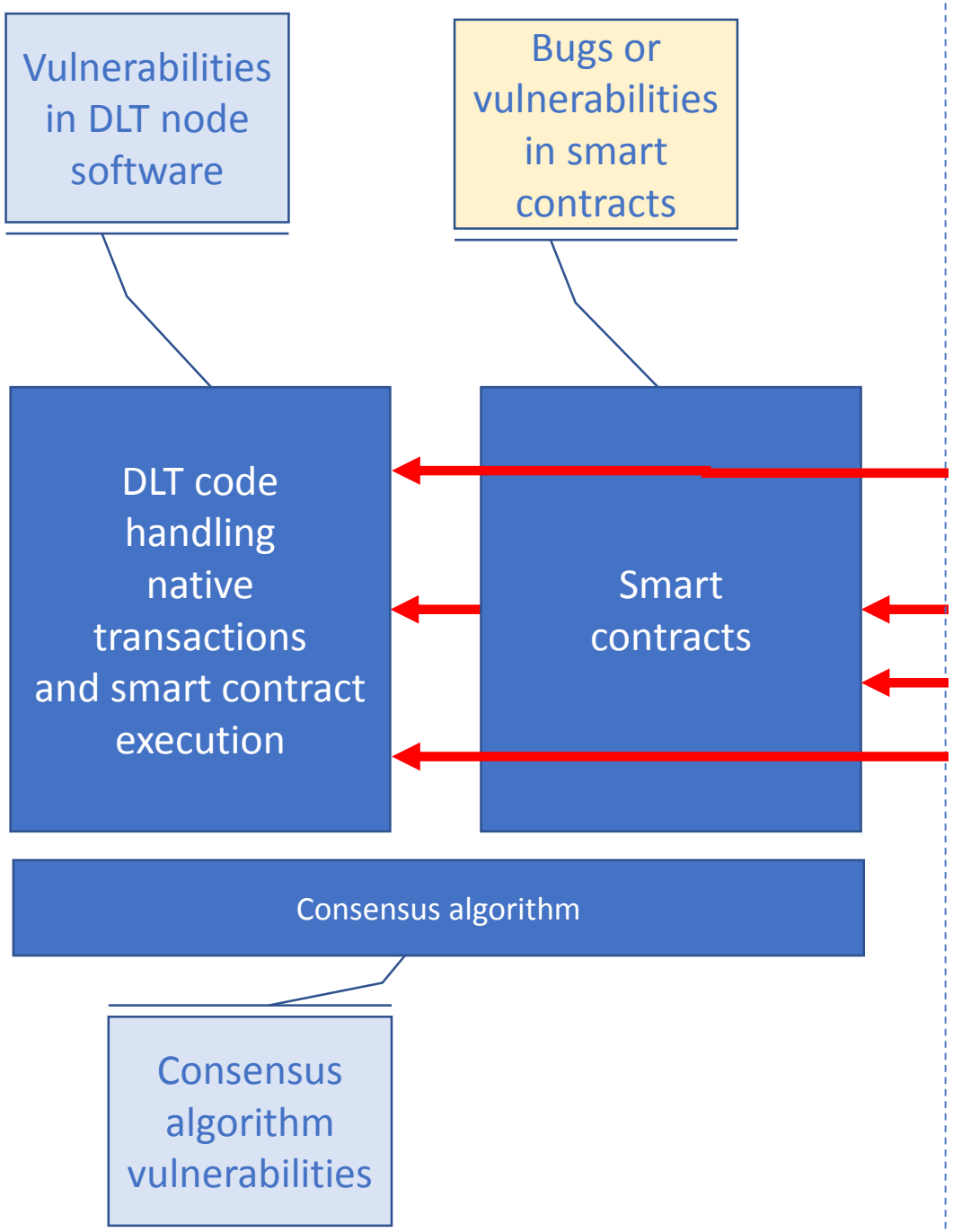
Responsibility: Designers and developers of smart contracts.



Blockchain-Side Vulnerabilities

Smart contract vulnerabilities

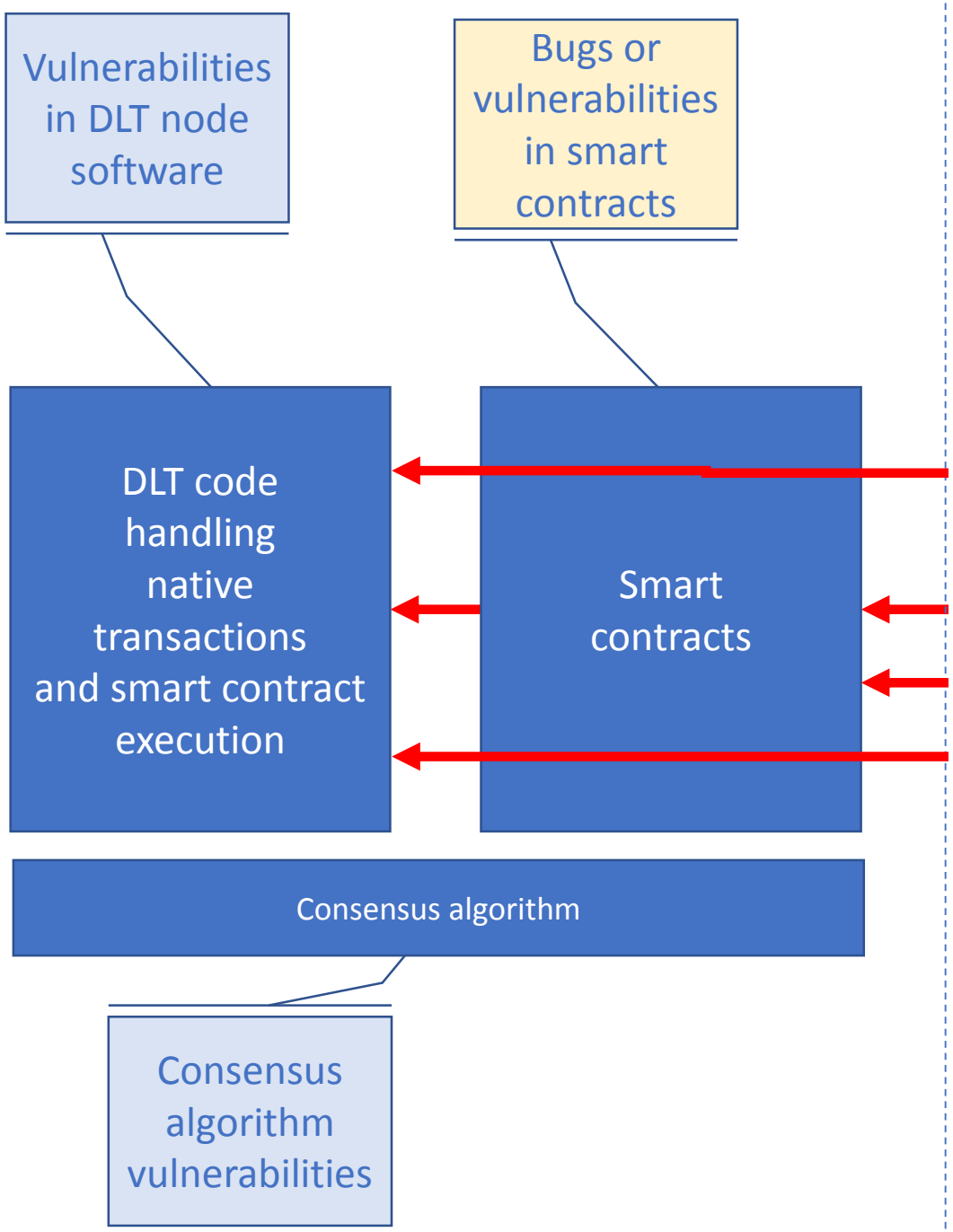
Bugs and deviations from intended functionality.



Blockchain-Side Vulnerabilities

Smart contract vulnerabilities
Bugs and deviations from intended functionality.

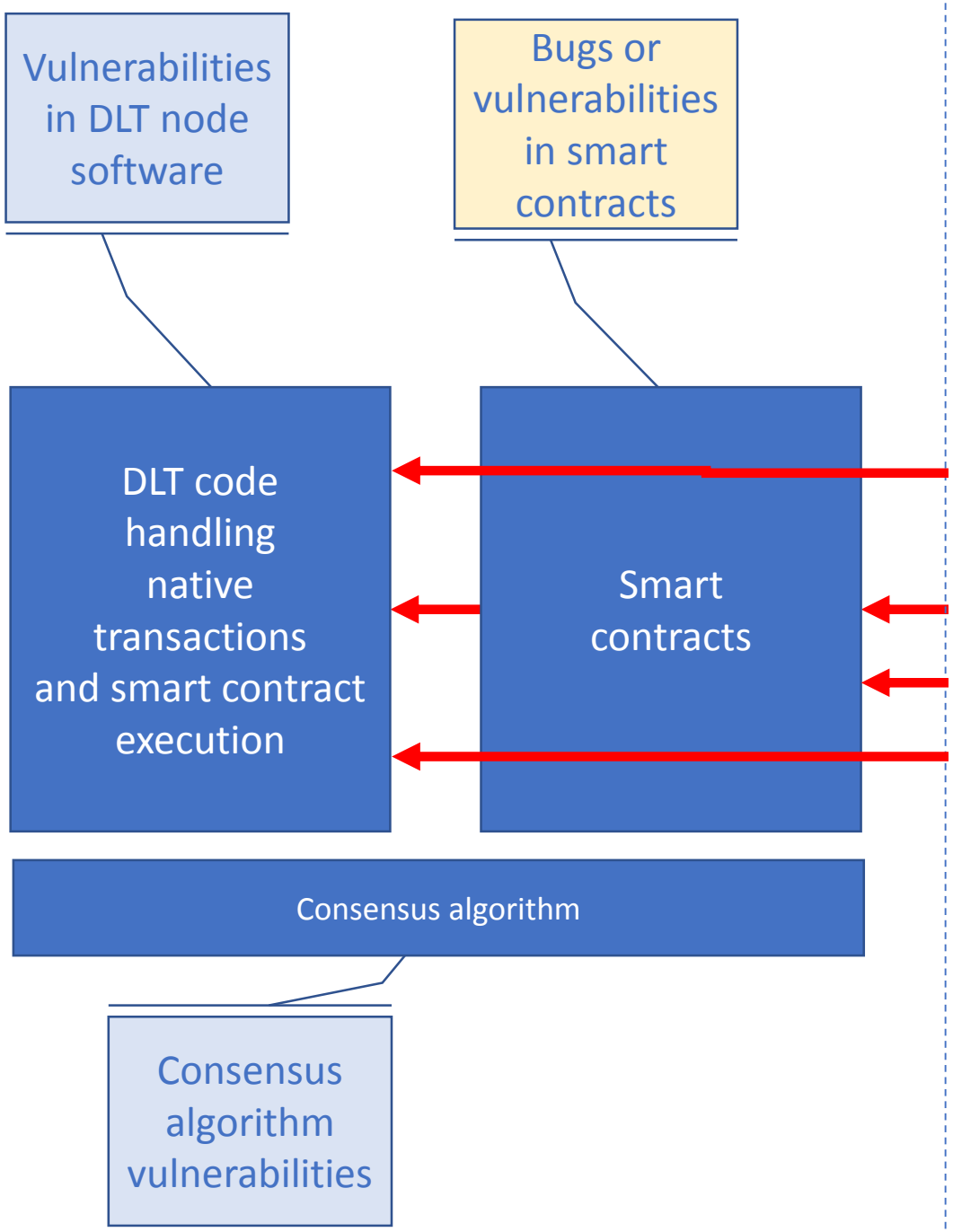
Have you ever used software that worked perfectly?



Blockchain-Side Vulnerabilities

Smart contract vulnerabilities
Bugs and deviations from intended functionality.

Have you ever used software that worked perfectly? Even worse...
Critical nature of smart contracts should require experienced software engineers using robust engineering practices...



Blockchain-Side Vulnerabilities

Smart contract vulnerabilities

Bugs and deviations from intended functionality.

Have you ever used software that worked perfectly? Even worse…

Critical nature of smart contracts should require experienced software engineers using robust engineering practices…

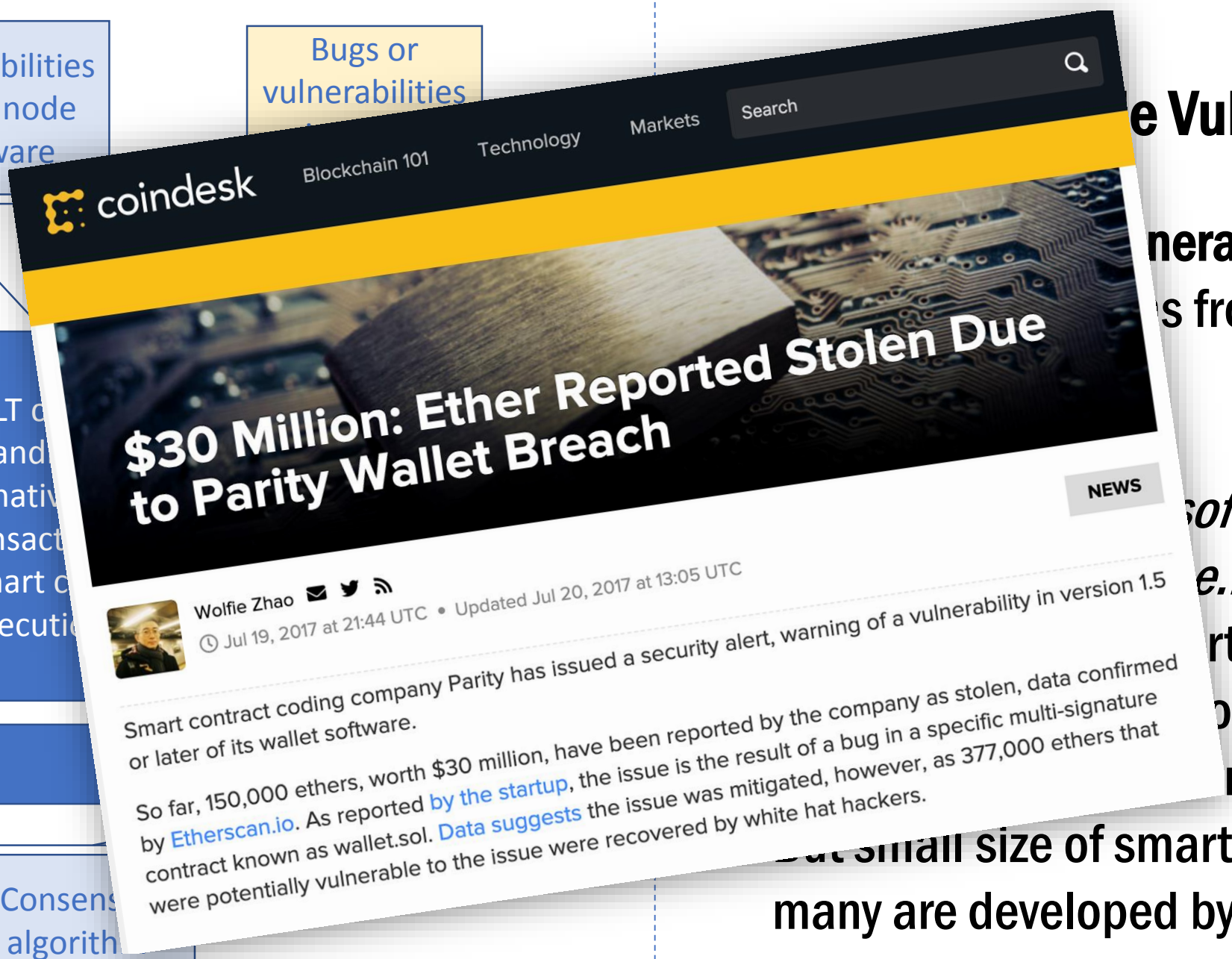
But small size of smart contracts means many are developed by inexperienced programmers.

Vulnerabilities in DLT node software

Bugs or vulnerabilities

DLT c
hand
nativ
transact
and smart c
executi

Consens
algorith
vulnerabilities



e Vulnerabilities

nerabilities
s from intended

oftware that worked
e...

rt contracts should
oftware engineers
ng practices...

But small size of smart contracts means many are developed by inexperienced programmers.

Vulnerabilities in DLT node software

Bugs or vulnerabilities

DLT c… hand… nativ… transact… and smart c… executi…

Consens… algorith… vulnerabilities

…e Vulnerabilities

…nerabilities

…intended

…rked

…ould

…eers

…

…neans

many are developed by inexperienced programmers.

Vulnerabilities in DLT node software

Bugs or vulnerabilities

DLT c… hand… nativ… transact… and smart c… executi…

Consens… algorith… vulnerabilities

e Vulnerabilities

…ded

…ational

…n

…rked

…ly …ould

…eers

…

…neans

…rienced

programmers.

Search

Market Cap: $273.47B | Reported Vol: $50.91B | Trusted Vol: $2.77B | BTC Dominance: 71.2% | Cryptocurrencies: 2,332

CRYPTOSLATE SIGN UP TRENDING NEWS COINS DIRECTORY PLACES NEW RESEARCH

EOS

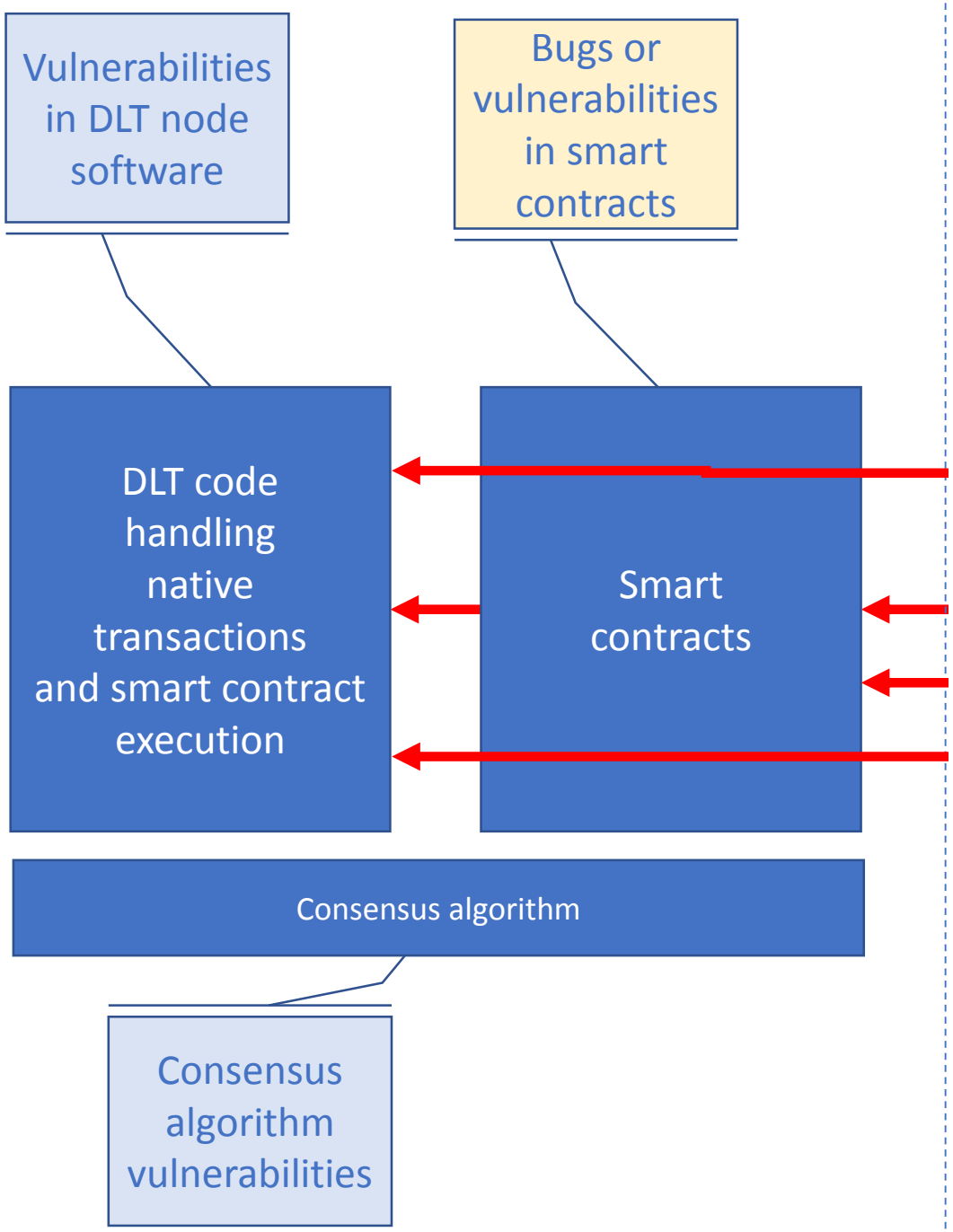
EOS DApp Smart Contract Exploit Pays Out $200K to Hacker

Sam Town · Sep 16, 2018 · 1 min read

Smart contract weaknesses in EOS-based gambling dApp EOBet have allowed hackers to manipulate the outcome of blockchain dice rolls, capturing 126,000 EOS in just 36 hours.

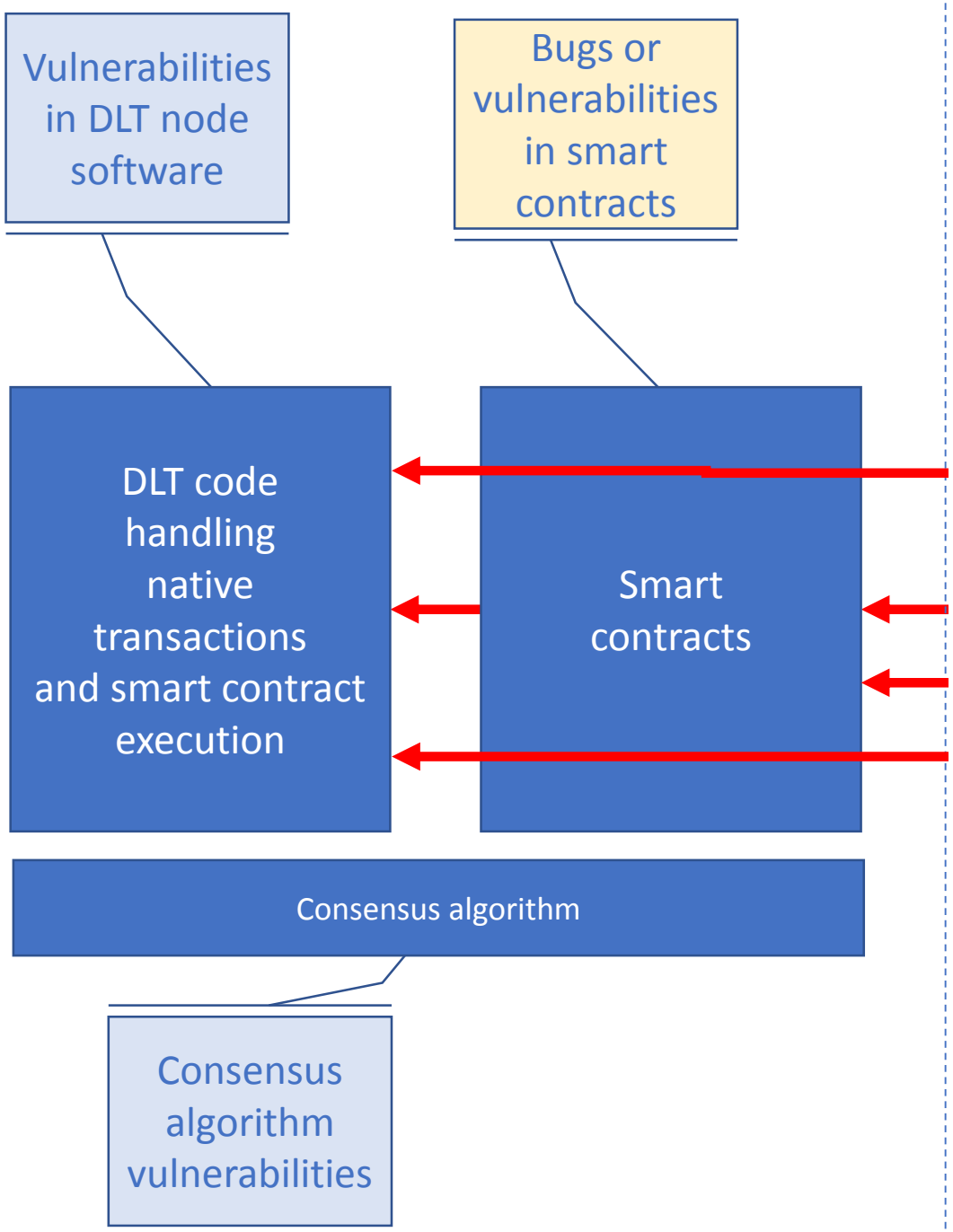
An official announcement from EOSBet explains the manner in which the attack was executed—by exploiting a flaw in smart contract code, the hacker was able to place bets without transferring EOS to the contract, while still capturing payouts from successful predictions.

were pote…



Blockchain-Side Vulnerabilities

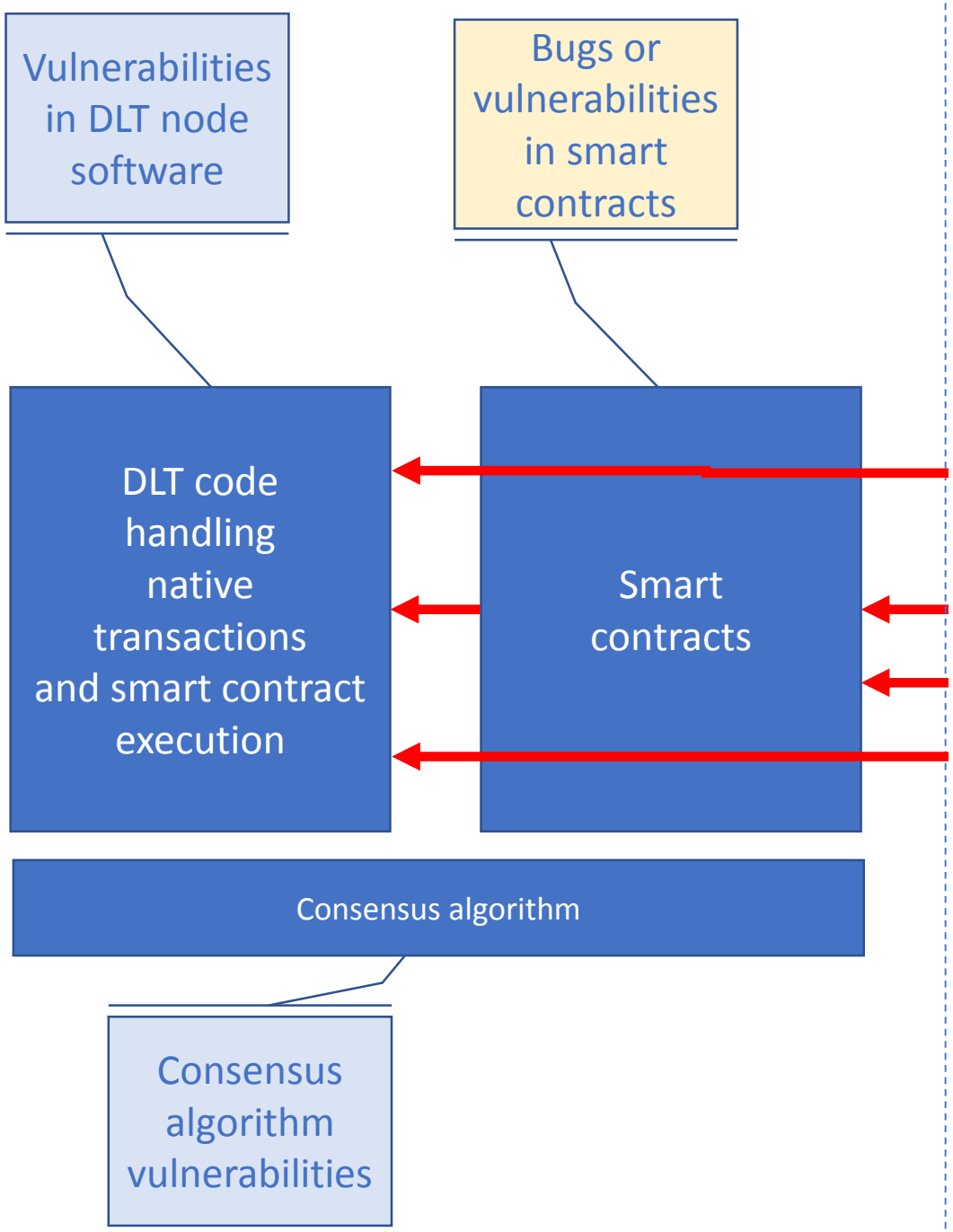
Smart contract vulnerabilities
Platform or programming language vulnerabilities.



Blockchain-Side Vulnerabilities

Smart contract vulnerabilities
Platform or programming language vulnerabilities.

The tools should deter, not foster mistakes.



Blockchain-Side Vulnerabilities

Smart contract vulnerabilities
Platform or programming language vulnerabilities.

The tools should deter, not foster mistakes.

Alas, few smart contract platforms and programming languages have been developed by the right experts.

Blockchain-Side Vulnerabilities

Vulnerabilities in DLT node software

Bugs or vu

DLT code handling native transactions and smart contract execution

Consensus a

Consensus algorithm vulnerabilities


BBC Sign in

News Sport Weather Shop Reel Travel

NEWS

Home Video World UK Business Tech Science Stories Entertainment & Arts

Technology

Hack attack drains start-up investment fund

21 June 2016

Share

Hackers have taken control of virtual cash worth $60m (£41m) by exploiting a bug in a system designed to help start-ups.

The attack targeted an investment fund called the DAO which is based on technology derived from the Bitcoin crypto-currency.

DAO members are now debating how to recover the diverted funds.

es

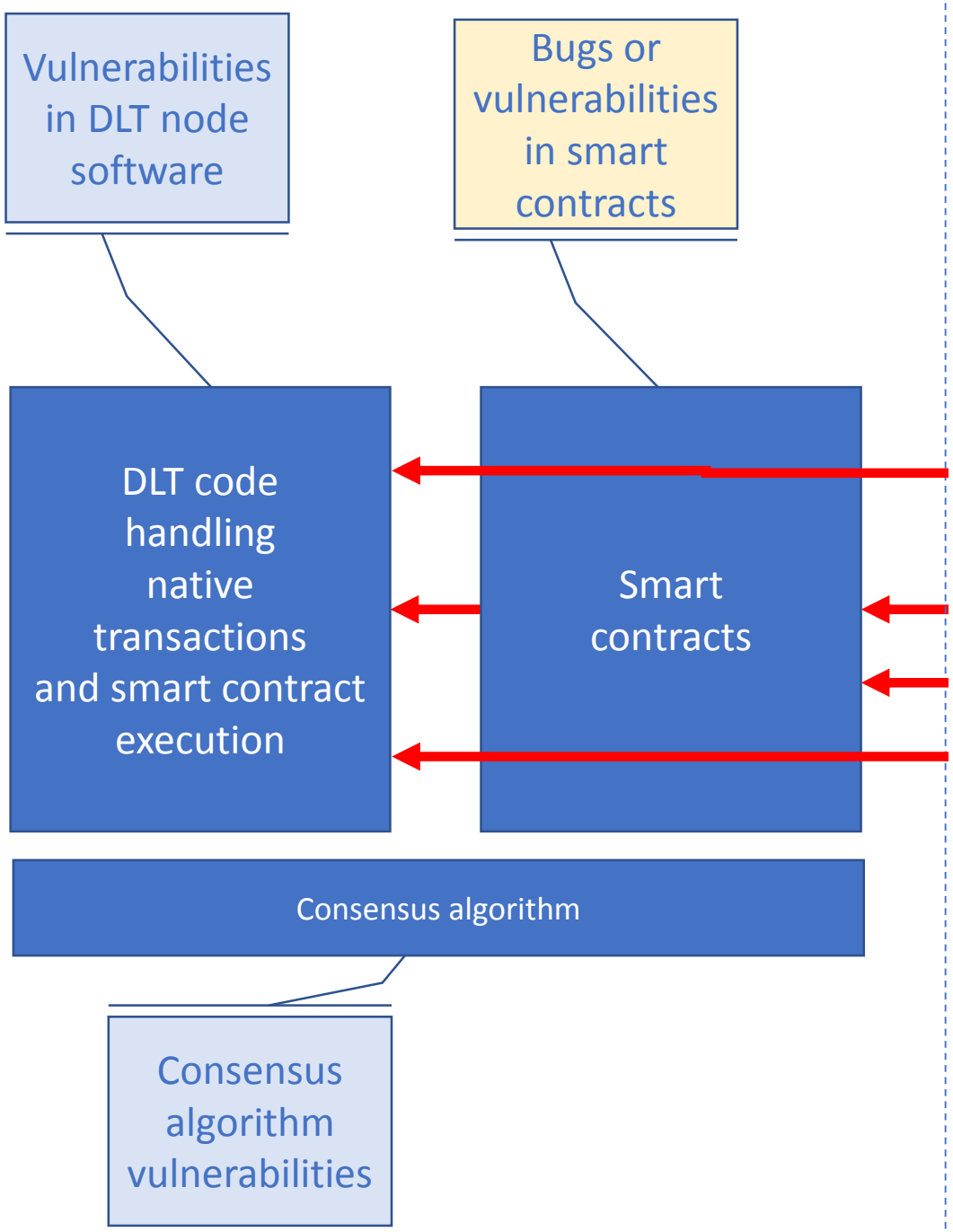
nguage

foster mistakes.

atforms and

ave been

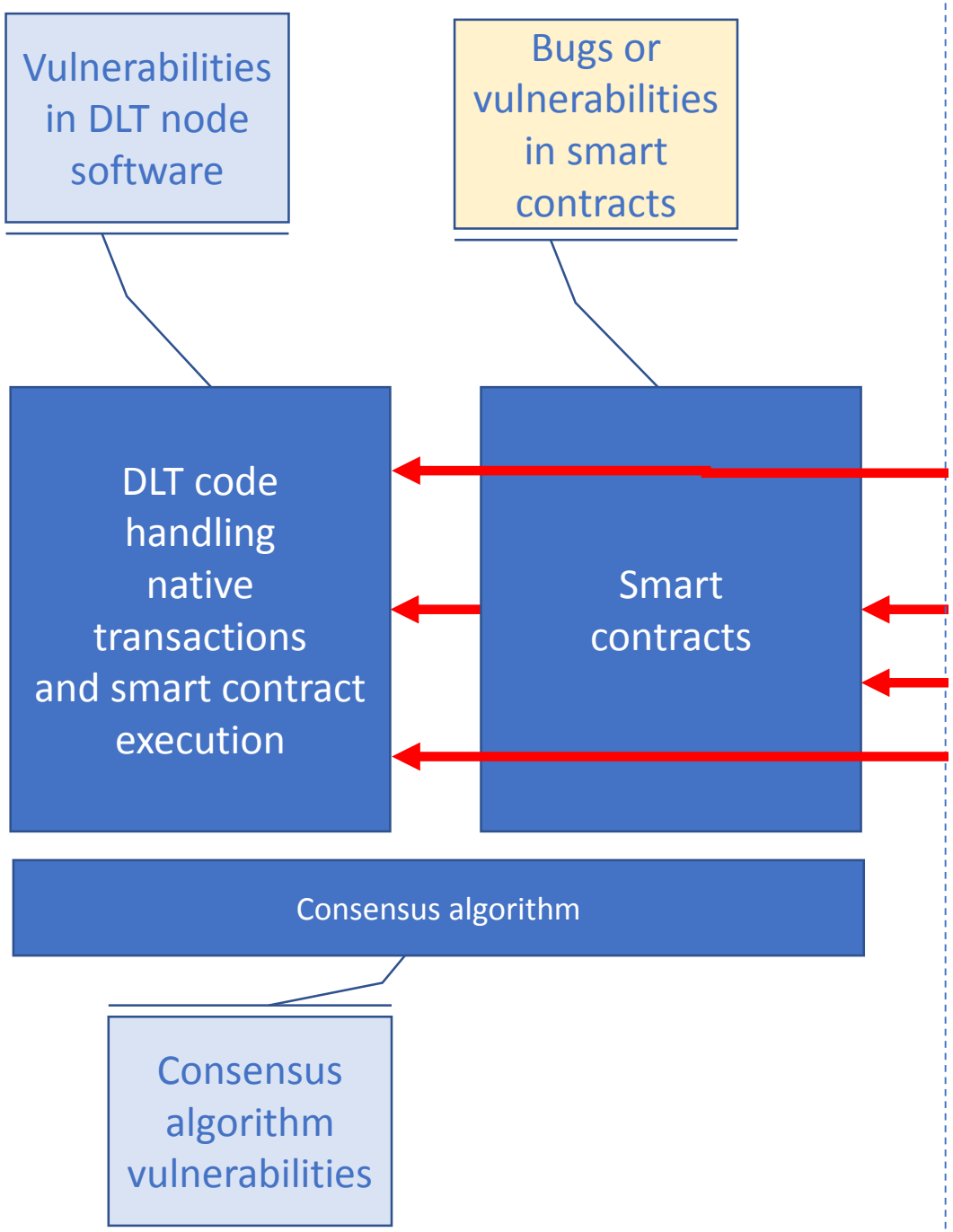
erts.



Blockchain-Side Vulnerabilities

Smart contract vulnerabilities

Malicious code.

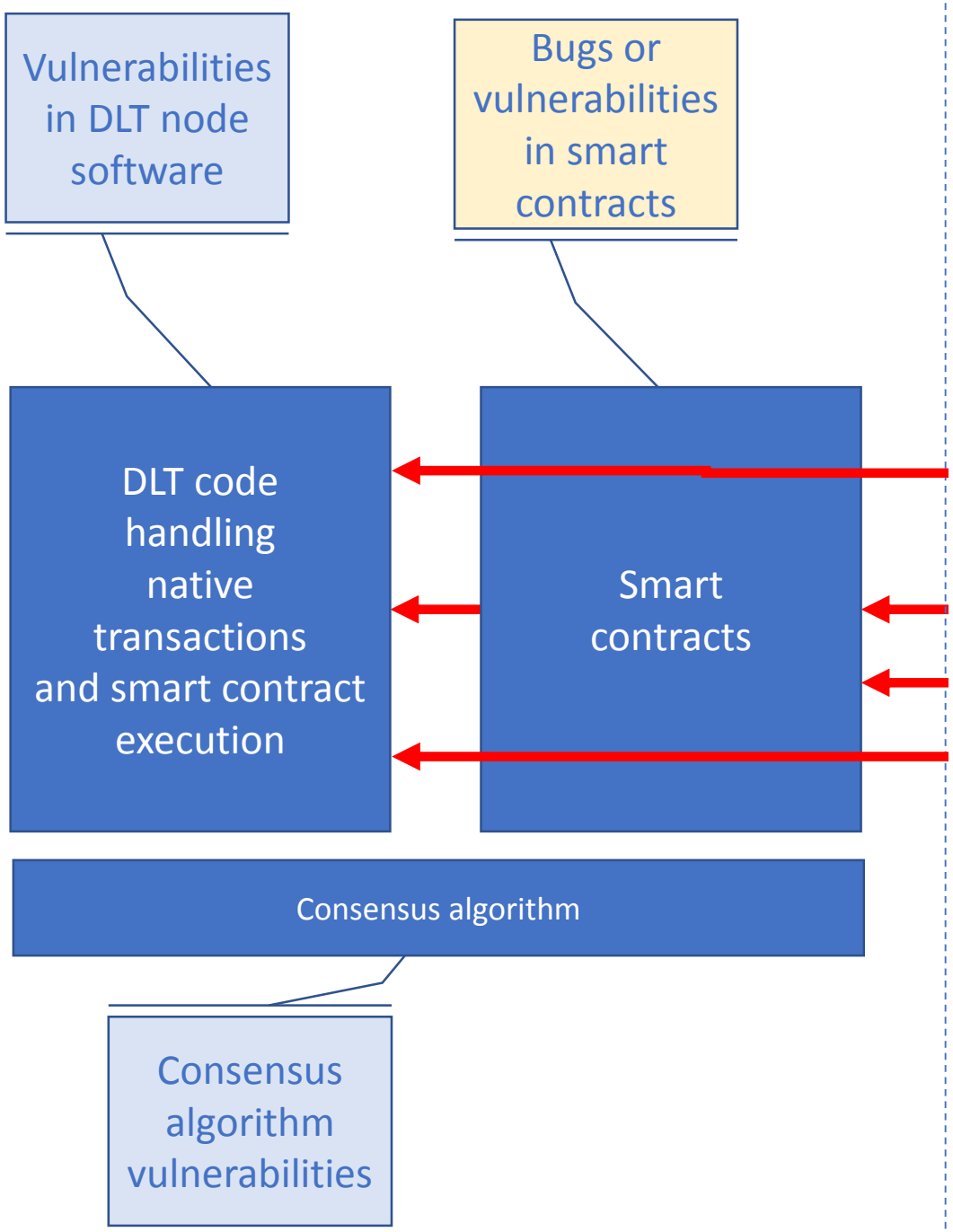


Blockchain-Side Vulnerabilities

Smart contract vulnerabilities

Malicious code.

Purposefully introducing obfuscated means of achieving unexpected behavior.



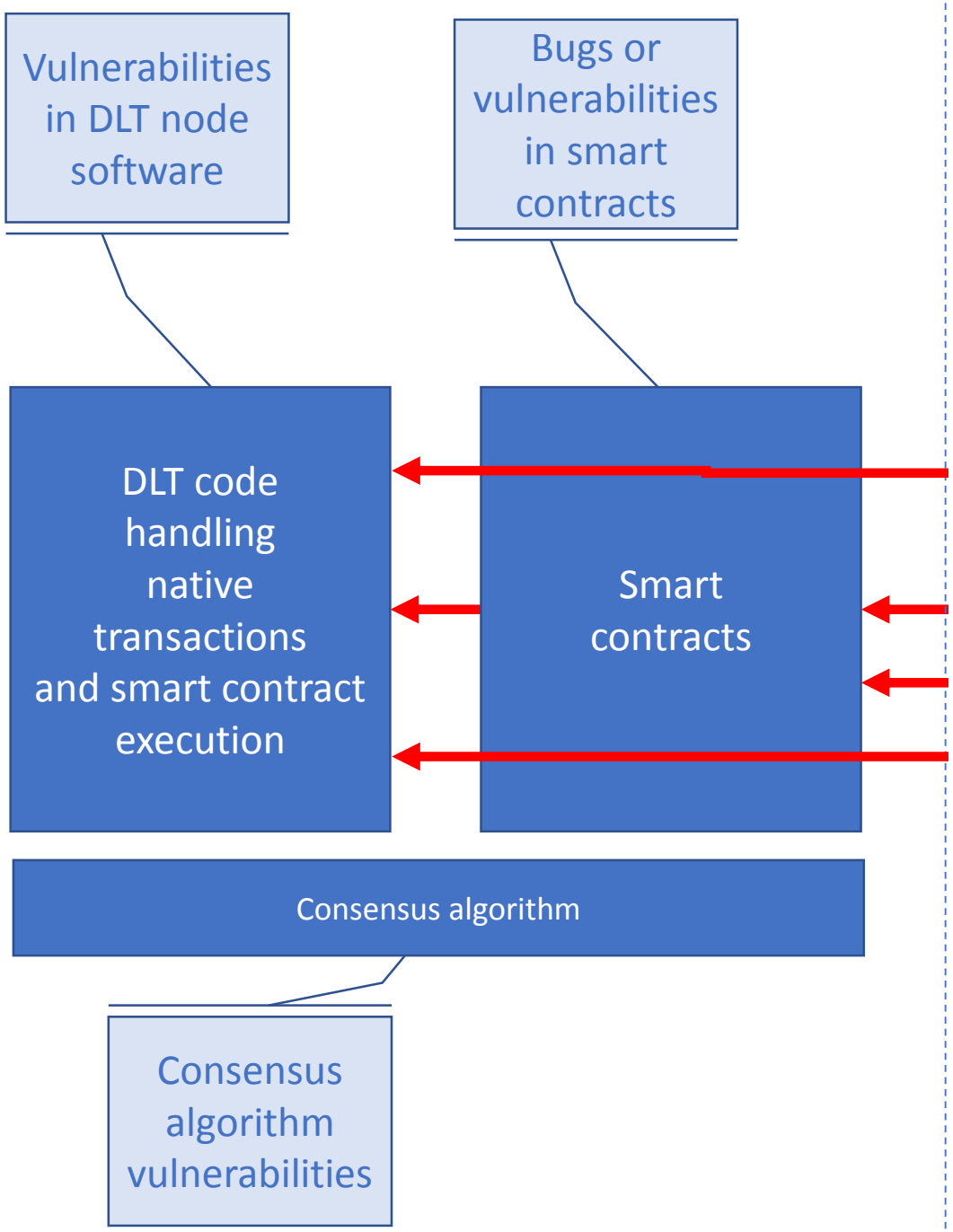
Blockchain-Side Vulnerabilities

Smart contract vulnerabilities

Malicious code.

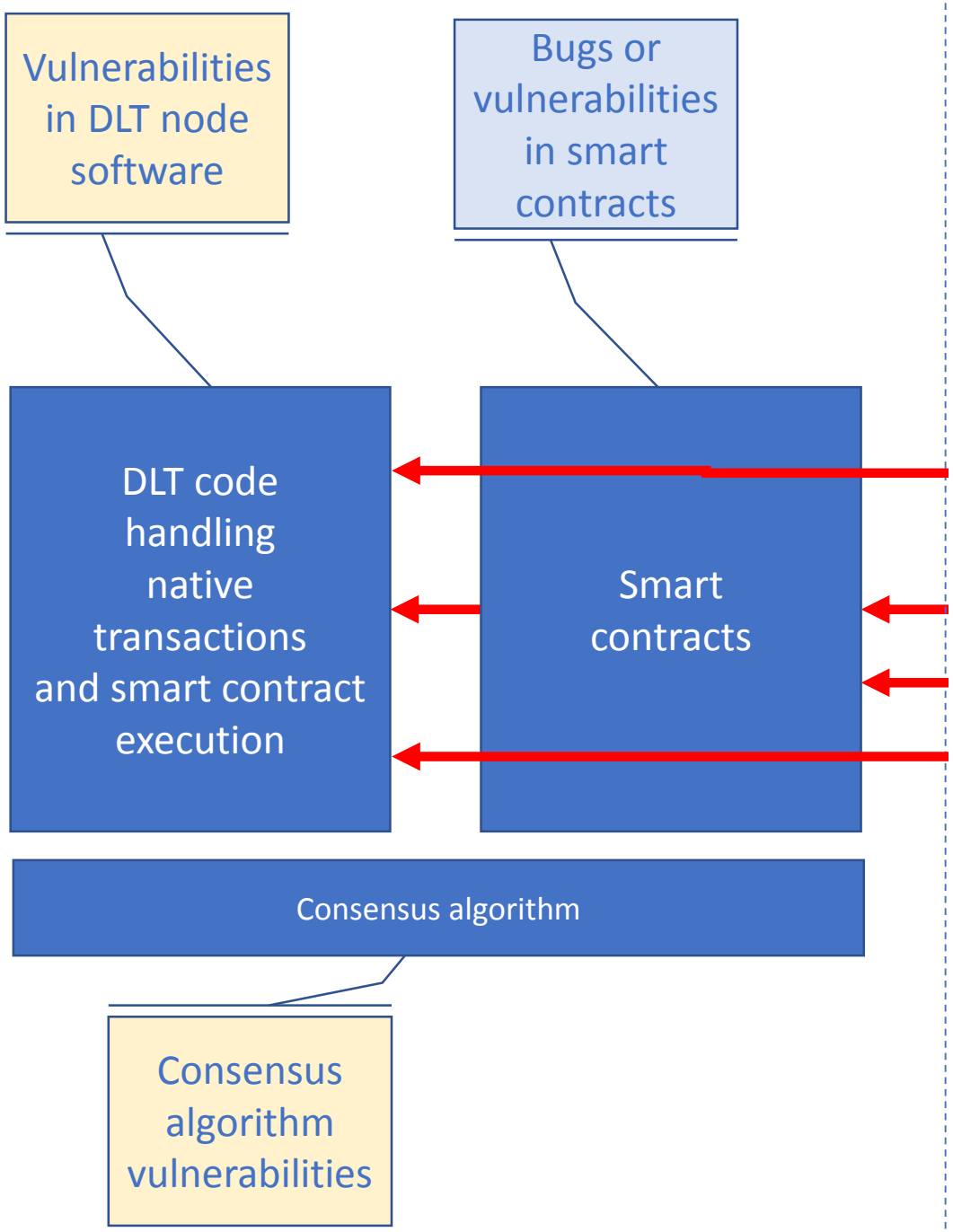
Purposefully introducing obfuscated means of achieving unexpected behavior.

Treat the developer as an adversary, not an ally.



Blockchain-Side Vulnerabilities

Hybrid Vulnerabilities



Blockchain-Side Vulnerabilities

7. Node software vulnerabilities

- Taking down nodes through known vulnerabilities to perform 51% attack.

Responsibility: DLT developers and miners.

Vulnerabilities in DLT node software

Bugs or vulnerabilities in smart contracts

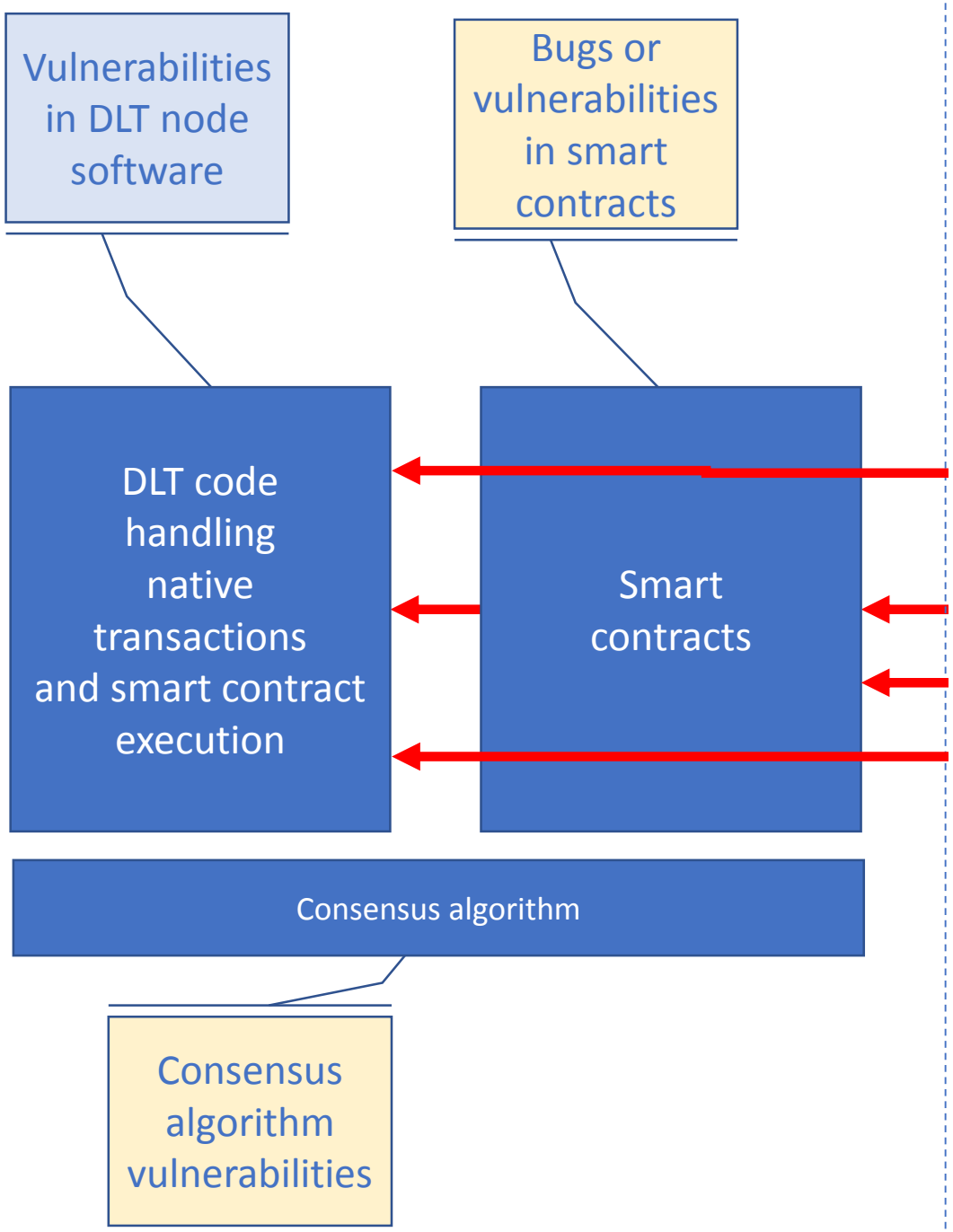
Blockchain-Side Vulnerabilities

Unpatched Ethereum Clients Pose 51% Attack Risk, Says Report

> "If a hacker can crash a large number of nodes, controlling 51% of the network becomes easier. Hence, software crashes are a serious security concern for blockchain nodes (unlike in other pieces of software where the hacker does not usually benefit from a crash)."

Consensus algorithm vulnerabilities

Responsibility: DLT developers and miners.



Blockchain-Side Vulnerabilities

8. Smart contracts and consensus

- Transaction reordering vulnerabilities.

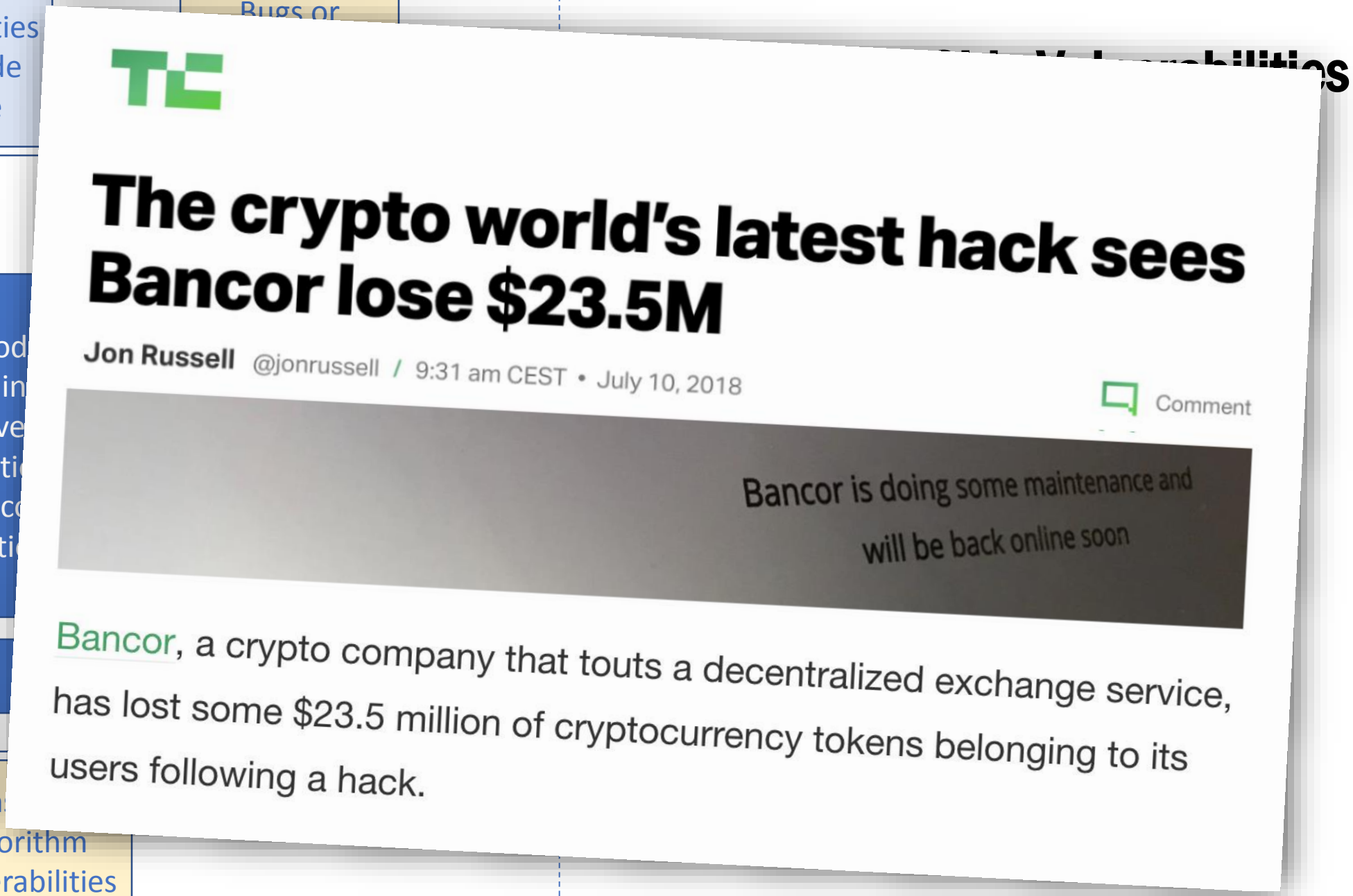
Responsibility: Smart contract designers.

Vulnerabilities in DLT node software

Bugs or

DLT cod
handlin
native
transacti
and smart c
executi

Con
algorithm
vulnerabilities



TC

The crypto world's latest hack sees Bancor lose $23.5M

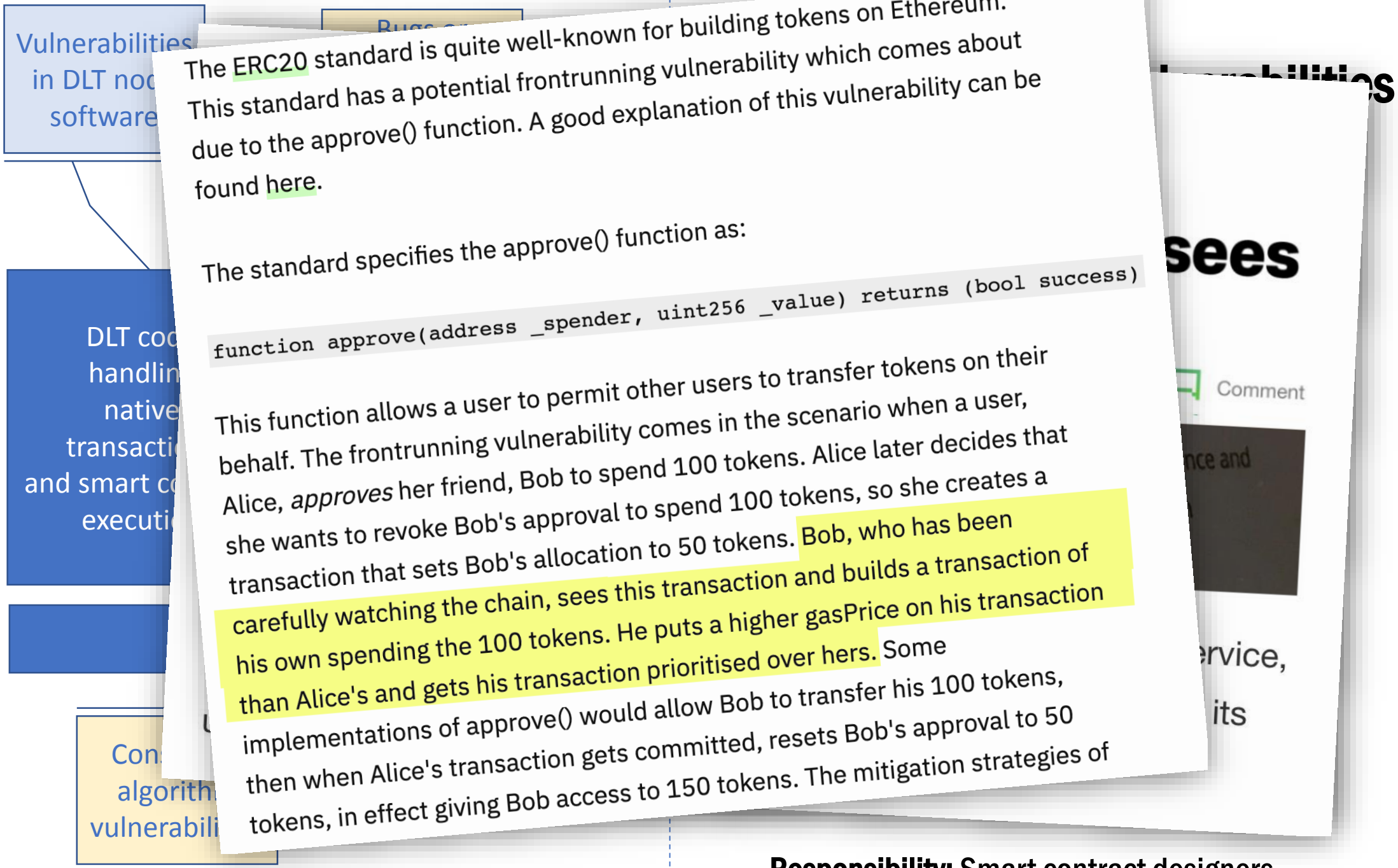
Jon Russell @jonrussell / 9:31 am CEST • July 10, 2018

Comment

Bancor is doing some maintenance and will be back online soon

Bancor, a crypto company that touts a decentralized exchange service, has lost some $23.5 million of cryptocurrency tokens belonging to its users following a hack.

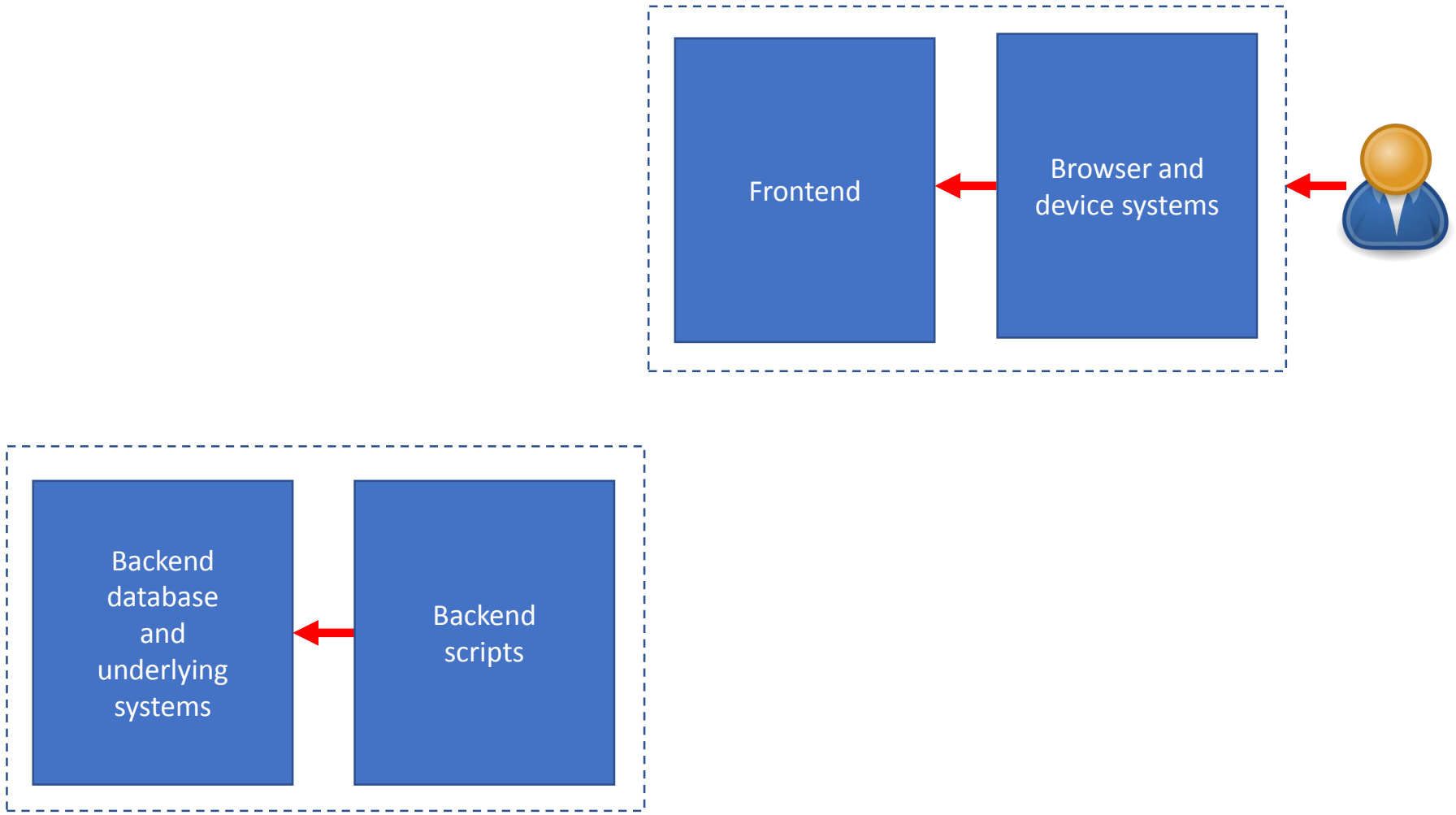
Responsibility: Smart contract designers.

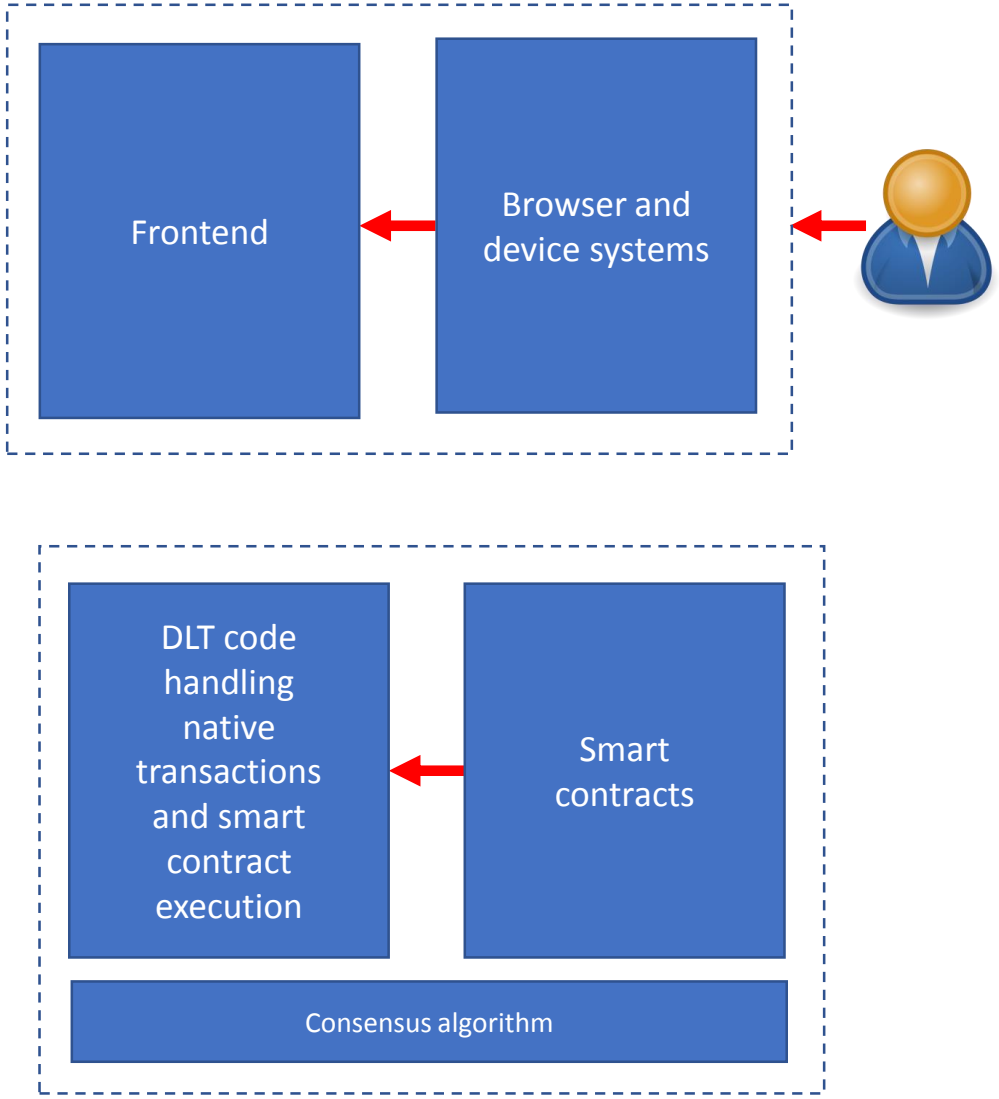


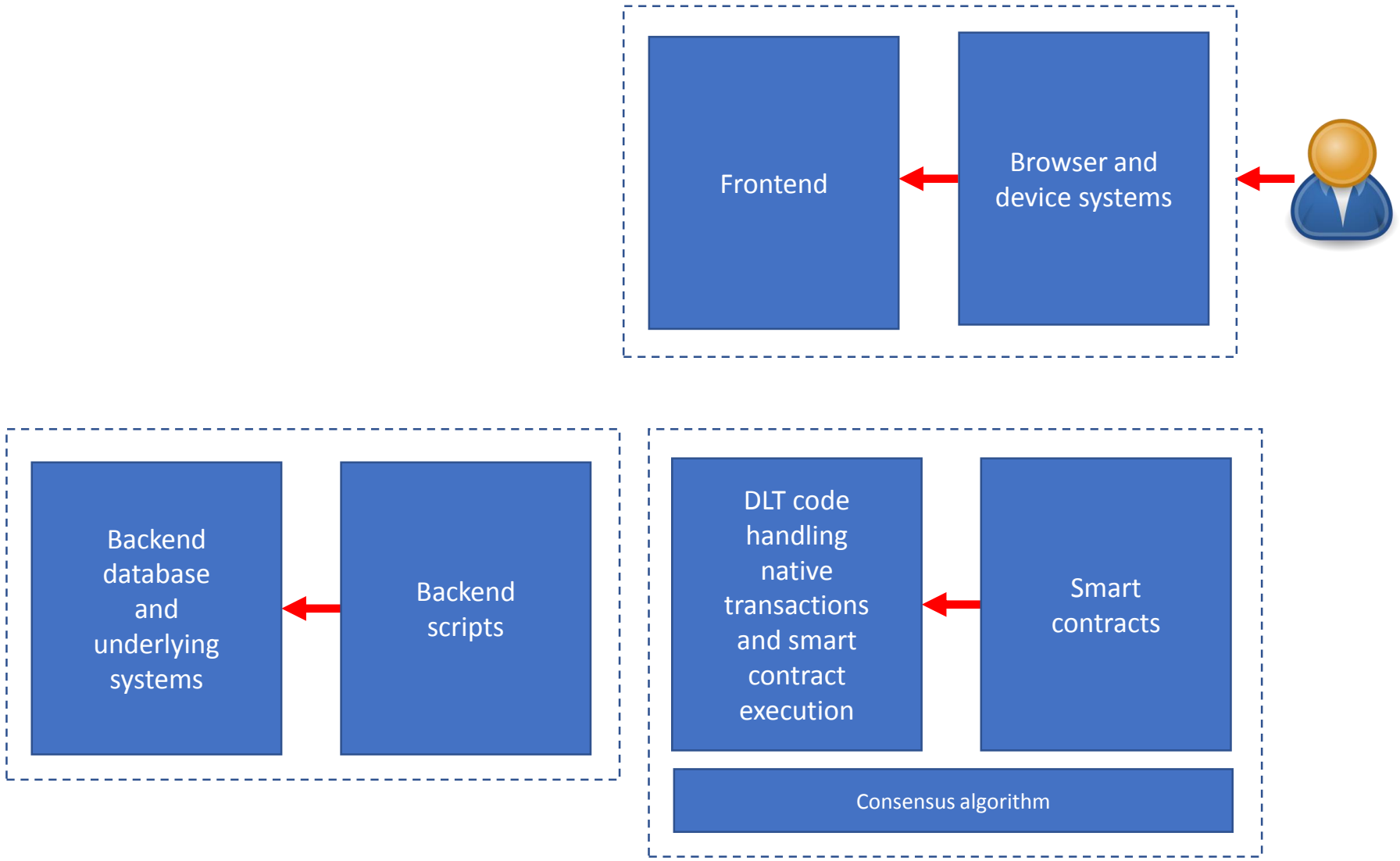
Responsibility: Smart contract designers.

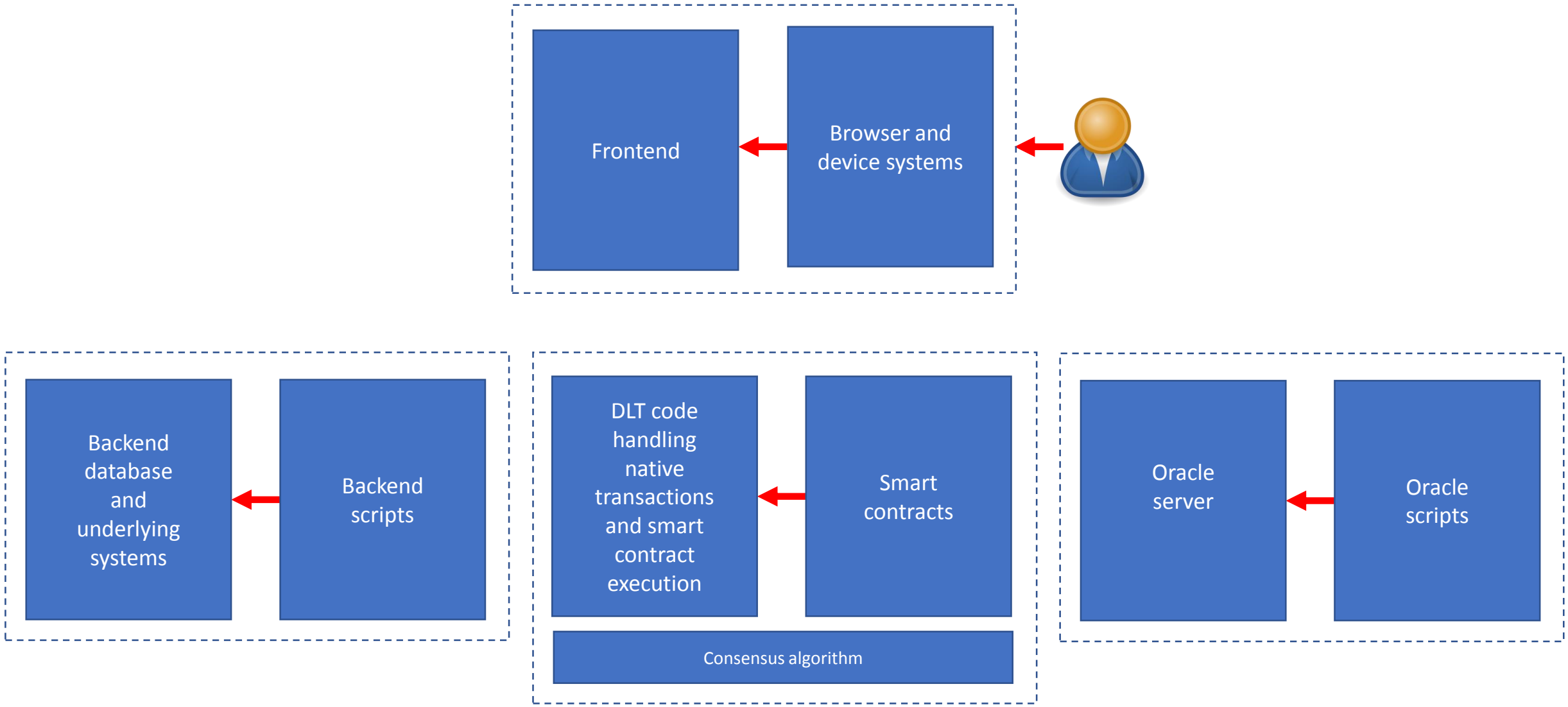
And this is but the tip of the iceberg…

- Various other known attack possibilities over the configurations shown.
- And these are but a subset of configurations…


Frontend
Browser and device systems
Backend database and underlying systems
Backend scripts


Frontend
Browser and device systems
DLT code handling native transactions and smart contract execution
Smart contracts
Consensus algorithm


Frontend
Browser and device systems
Backend database and underlying systems
Backend scripts
DLT code handling native transactions and smart contract execution
Smart contracts
Consensus algorithm

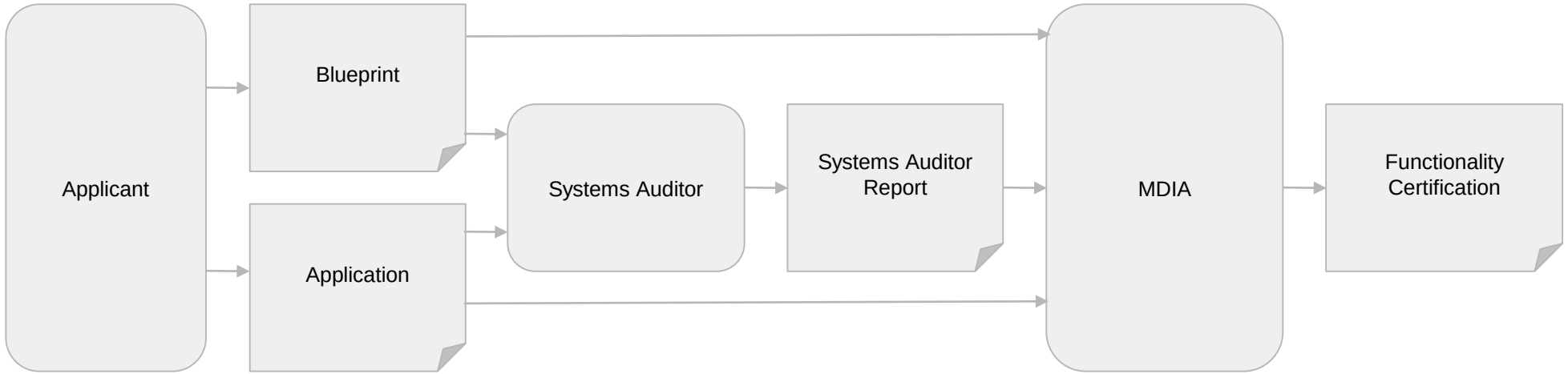

Frontend
Browser and device systems
Backend database and underlying systems
Backend scripts
DLT code handling native transactions and smart contract execution
Smart contracts
Consensus algorithm
Oracle server
Oracle scripts

And this is but the tip of the iceberg…

- Various other known attack possibilities over the configurations shown.
- And these are but a subset of configurations…
- **So what can we do to address these risks?**

Malta Digital Innovation Authority
Increasing User Trust and Protection in DLTs

- Independent Systems Audit approach being taken by the Malta Digital Innovation Authority.
- DLTs and smart contracts may be registered with the MDIA.



Malta Digital Innovation Authority
Increasing User Trust and Protection in DLTs

- The MDIA does not evaluate the technology itself, but rather licenses Systems Auditors who can do so.
- Their role is to apply due diligence to the technology, ensuring that:
  - Appropriate system design and architecture
  - Experience and processes in the development team
  - Appropriate security measures in place
  - Appropriate quality assurance processes
  - The software really functions as documented in the blueprint
  - Appropriate measures in place as required by the law and MDIA
  - Subject matter experts appointed by the SAs sign off each report section, with a concrete opinion about the ITA.

Malta Digital Innovation Authority
Increasing User Trust and Protection in DLTs

- Certified Innovative Technology Arrangements provide higher levels of assurance:
  - The claimed functionality is documented in the blueprint in English.
  - Applicant due diligence will be carried out by MDIA.
  - Technical due diligence provided by Systems Auditor's report.
  - Identifiable known legal entity (the applicant) who can be held liable.
  - Information is stored on a Forensic Node in case investigation is required.
  - There is some degree of power-of-intervention which may be requested from a Technical Administrator.
  - English description in the blueprint prevails over implementation.

Malta Digital Innovation Authority

www.mdia.gov.mt

gordon.pace@um.edu.mt