RSAConference2020

San Francisco | February 24 – 28 | Moscone Center

HUMAN ELEMENT

SESSION ID: **ACB-RO2**

# Blockchain and distributed ledger technologies
# Security risks, threats and vulnerabilities

**Kurt Callewaert**

Researchmanager Applied Computer Sciences
HOWEST UNIVERSITY
@KurtCallewaert
Kurt.Callewaert@howest.be

#RSAC

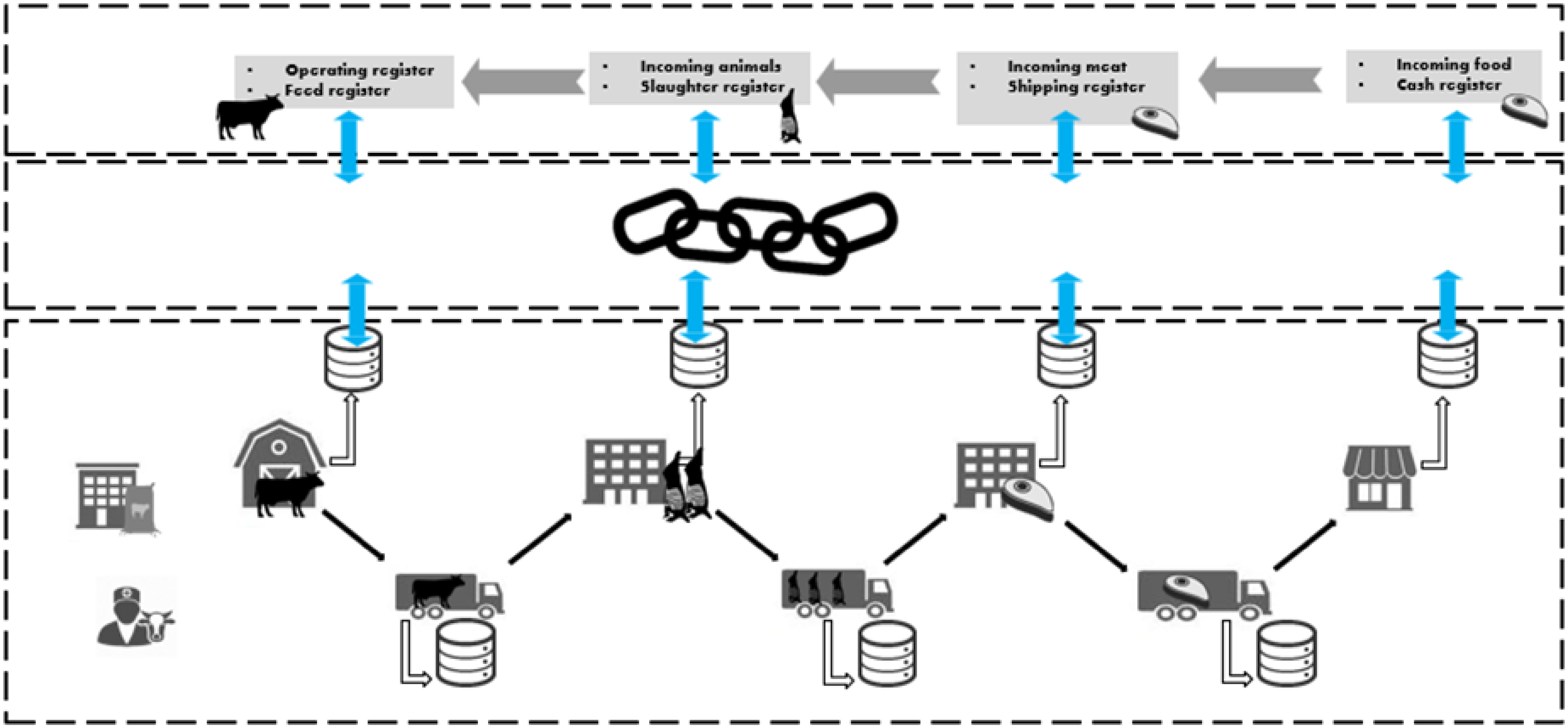
# Blockchain and distributed ledger technologies

# Are there still security risks, threats and vulnerabilities ?

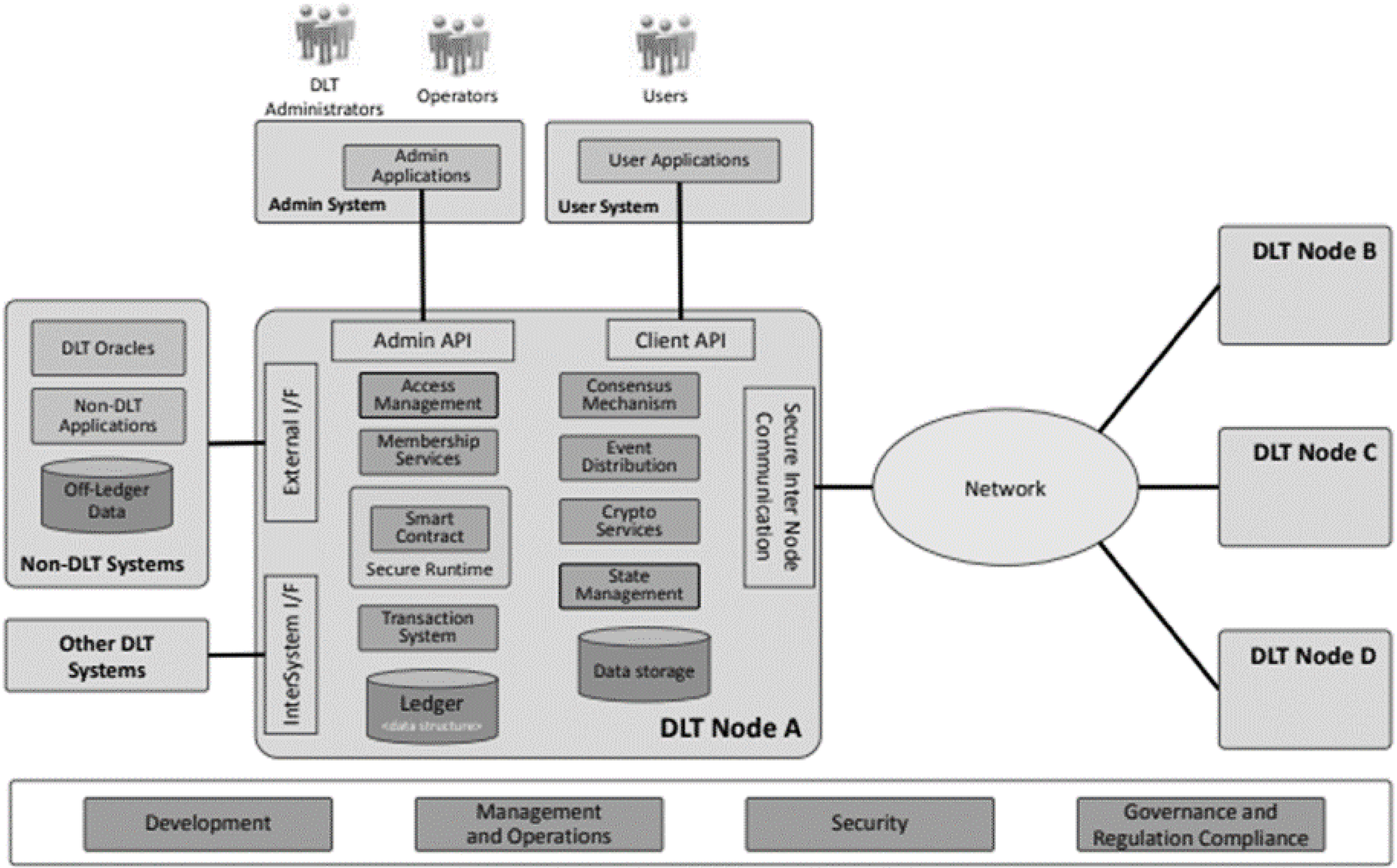
RSA Conference2020

# Building blocks in Blockchain & DLT technology

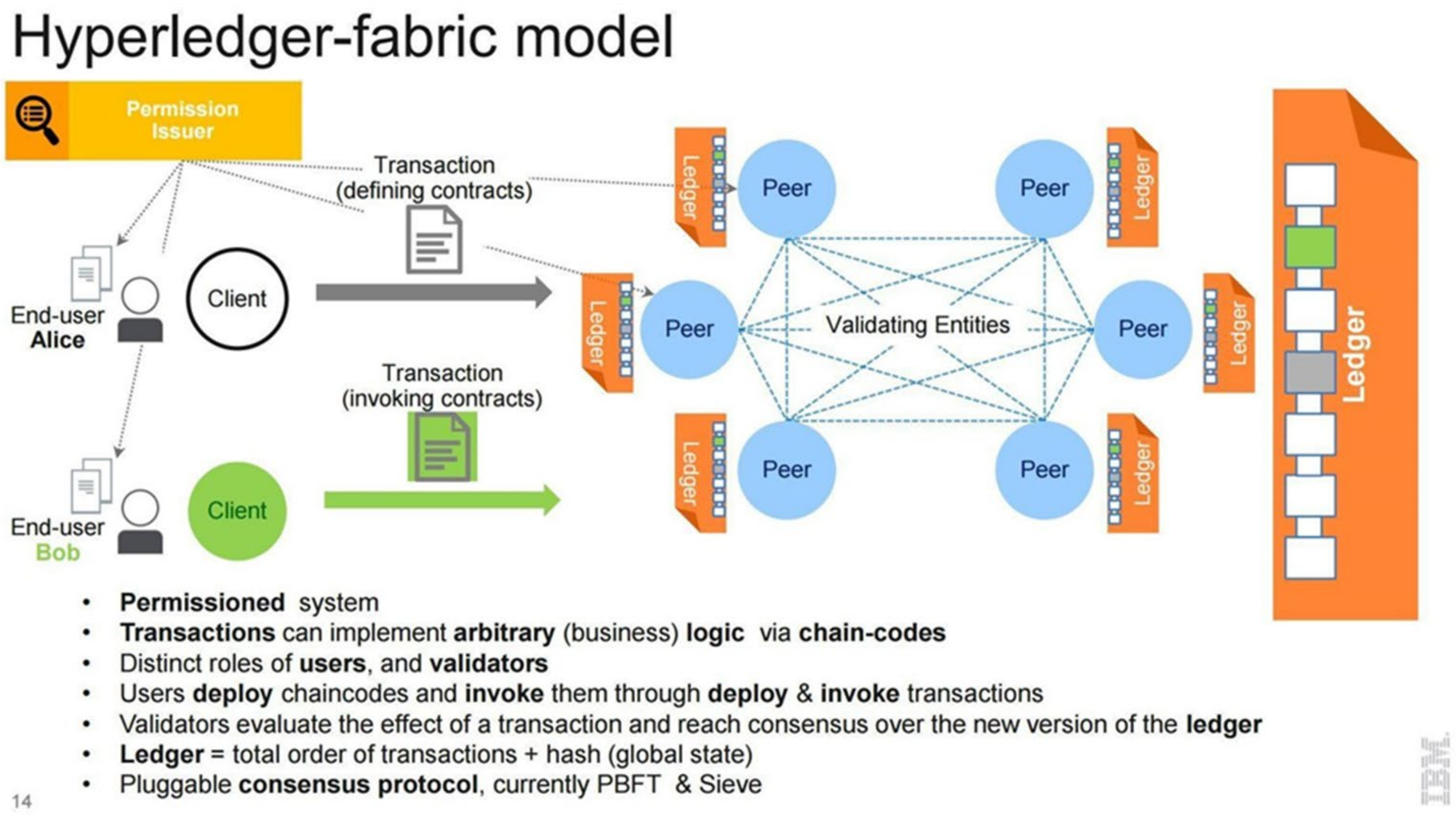
# Blockchain & Tracebility in the foodsector



# Blockchain functional view architecture



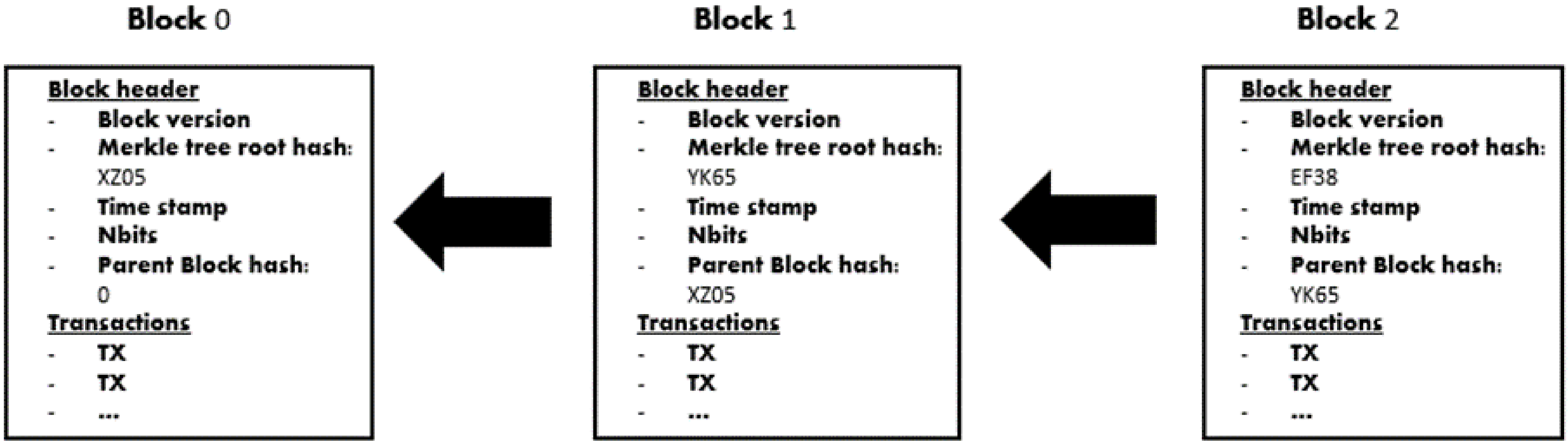
# Blockchain Model



## Hyperledger-fabric model

- **Permissioned** system
- **Transactions** can implement **arbitrary** (business) **logic** via **chain-codes**
- Distinct roles of **users**, and **validators**
- Users **deploy** chaincodes and **invoke** them through **deploy** & **invoke** transactions
- Validators evaluate the effect of a transaction and reach consensus over the new version of the **ledger**
- **Ledger** = total order of transactions + hash (global state)
- Pluggable **consensus protocol**, currently PBFT & Sieve

# Example block



**Block** 0

**Block header**
- **Block version**
- **Merkle tree root hash:** XZ05
- **Time stamp**
- **Nbits**
- **Parent Block hash:** 0

**Transactions**
- **TX**
- **TX**
- **...**

**Block** 1

**Block header**
- **Block version**
- **Merkle tree root hash:** YK65
- **Time stamp**
- **Nbits**
- **Parent Block hash:** XZ05

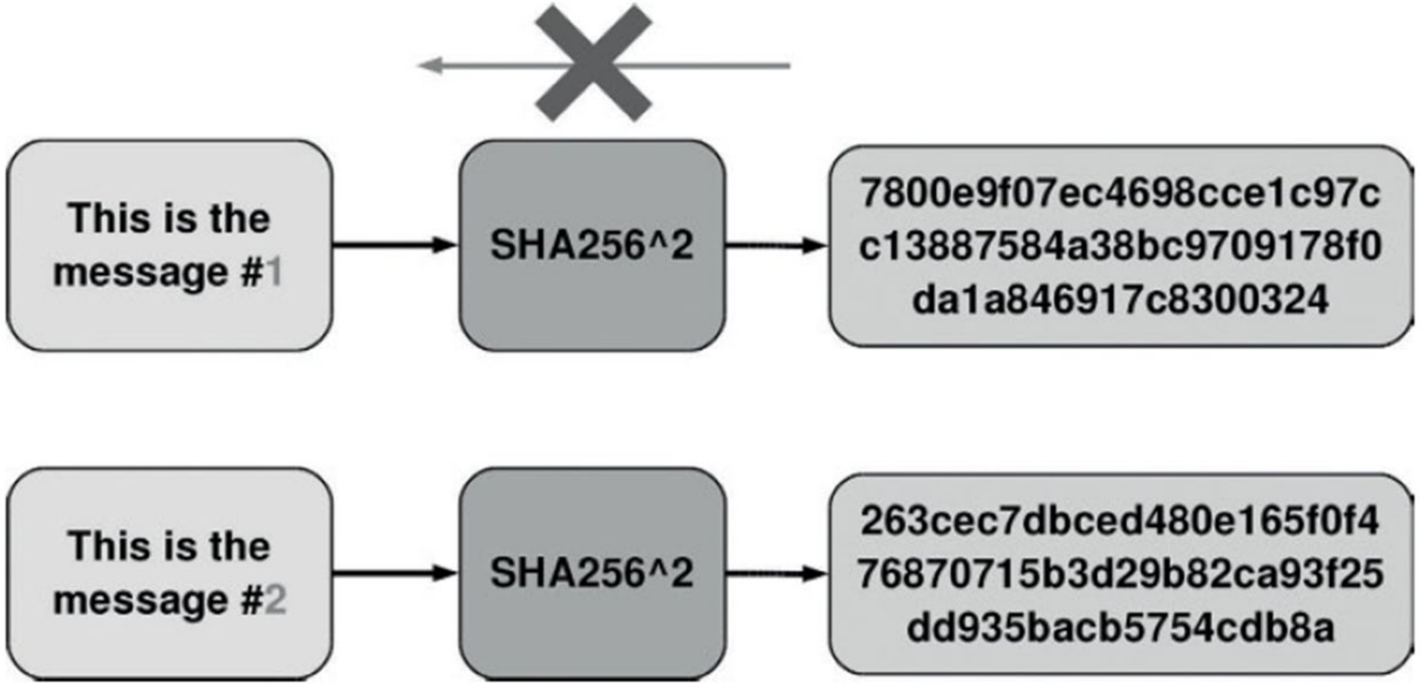
**Transactions**
- **TX**
- **TX**
- **...**

**Block** 2

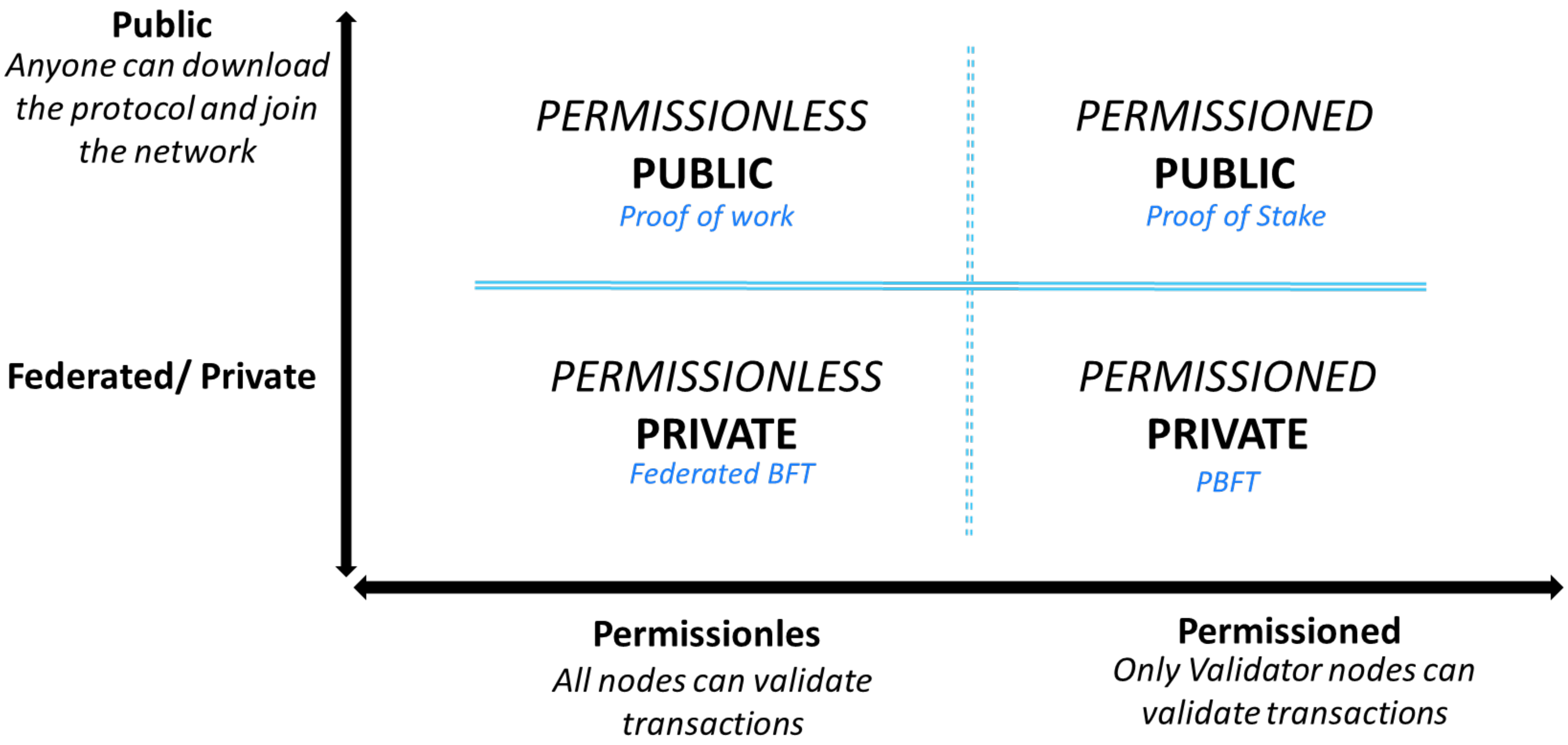
**Block header**
- **Block version**
- **Merkle tree root hash:** EF38
- **Time stamp**
- **Nbits**
- **Parent Block hash:** YK65

**Transactions**
- **TX**
- **TX**
- **...**

# Hashing


This is the message #1
SHA256^2
7800e9f07ec4698cce1c97c
c13887584a38bc9709178f0
da1a846917c8300324
This is the message #2
SHA256^2
263cec7dbced480e165f0f4
76870715b3d29b82ca93f25
dd935bacb5754cdb8a

# Blockchain implementations



**Public**
*Anyone can download the protocol and join the network*

**Federated/ Private**

*PERMISSIONLESS*
**PUBLIC**
*Proof of work*

*PERMISSIONED*
**PUBLIC**
*Proof of Stake*

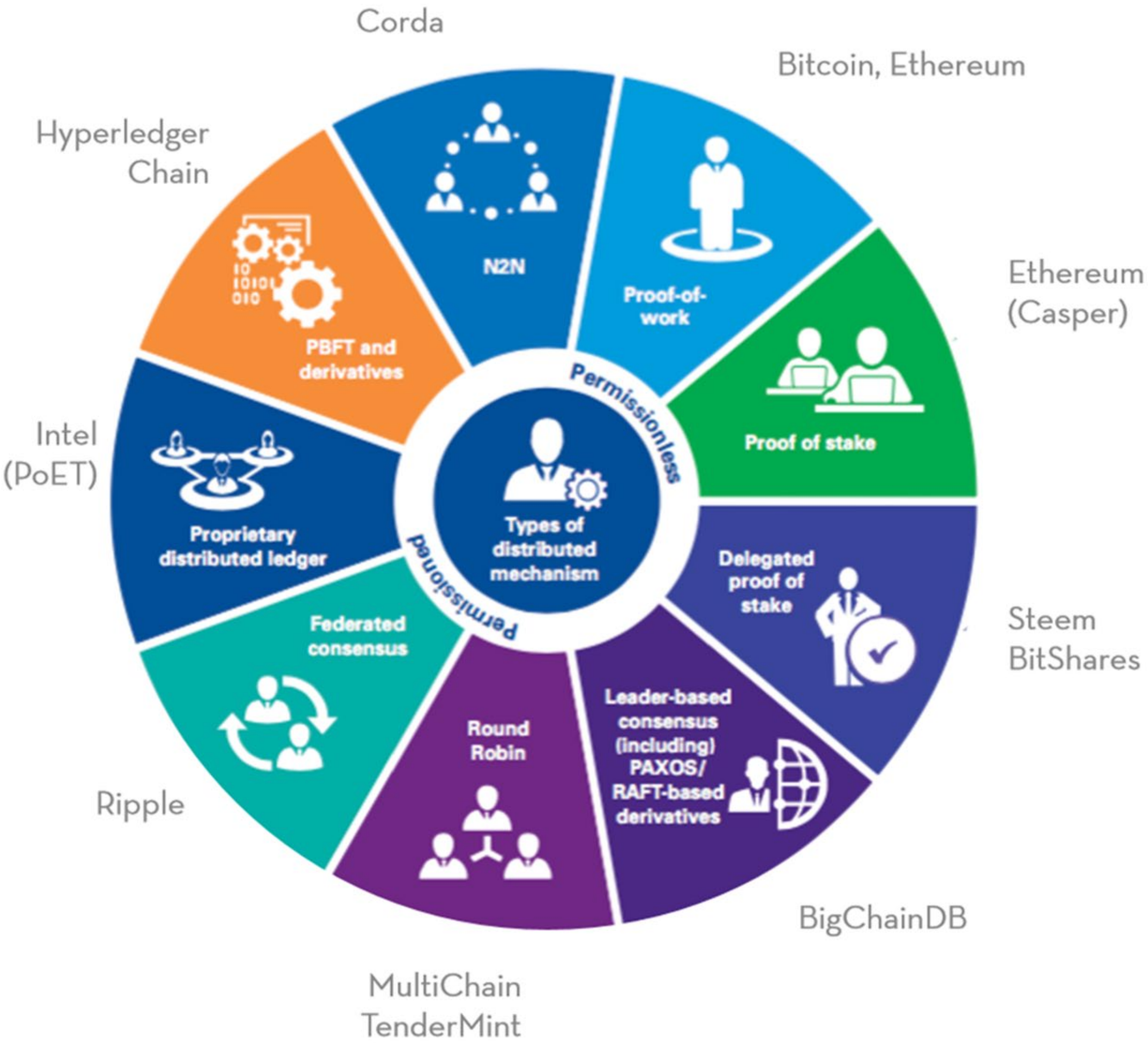
*PERMISSIONLESS*
**PRIVATE**
*Federated BFT*

*PERMISSIONED*
**PRIVATE**
*PBFT*

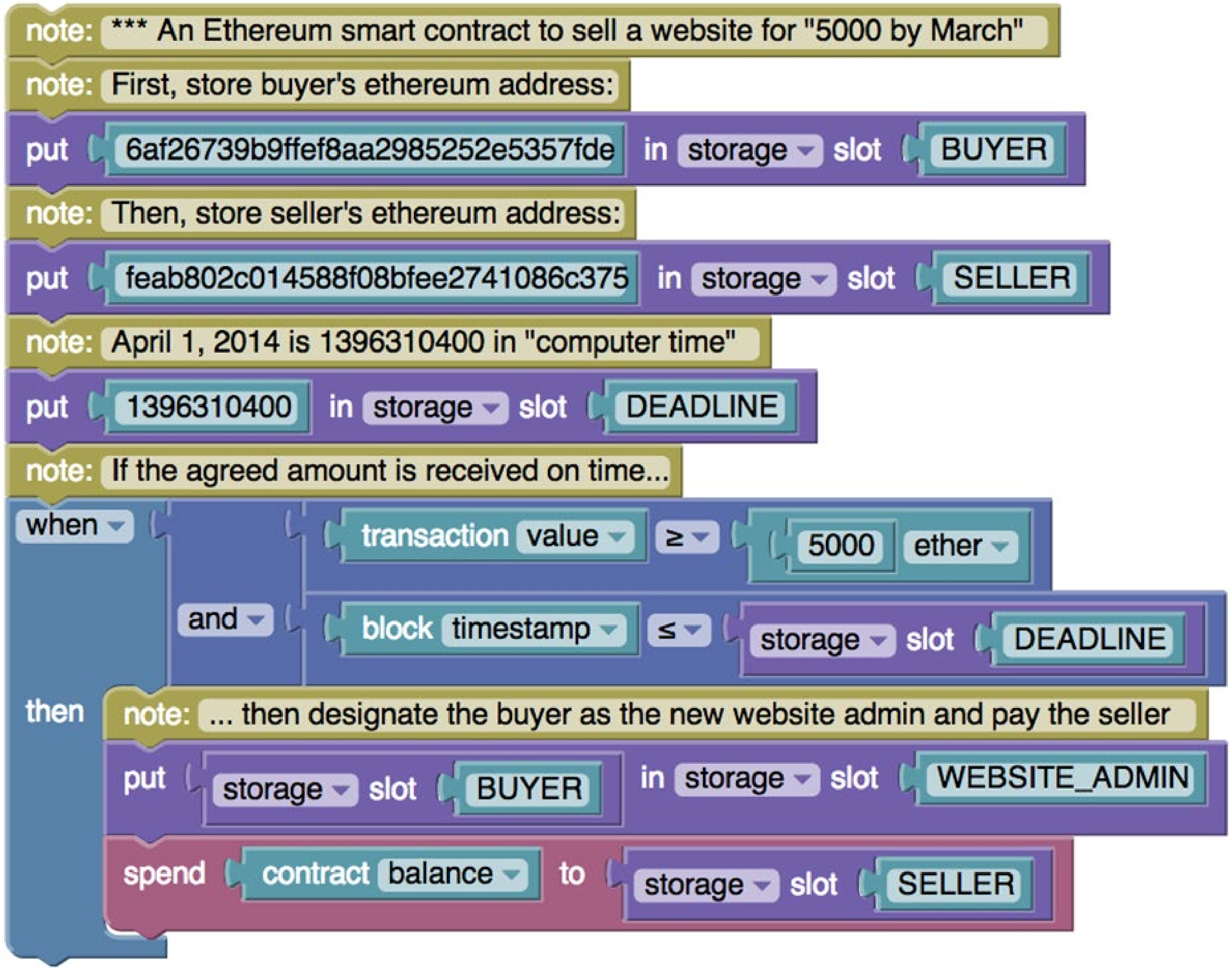
**Permissionles**
*All nodes can validate transactions*

**Permissioned**
*Only Validator nodes can validate transactions*

# Consensus model

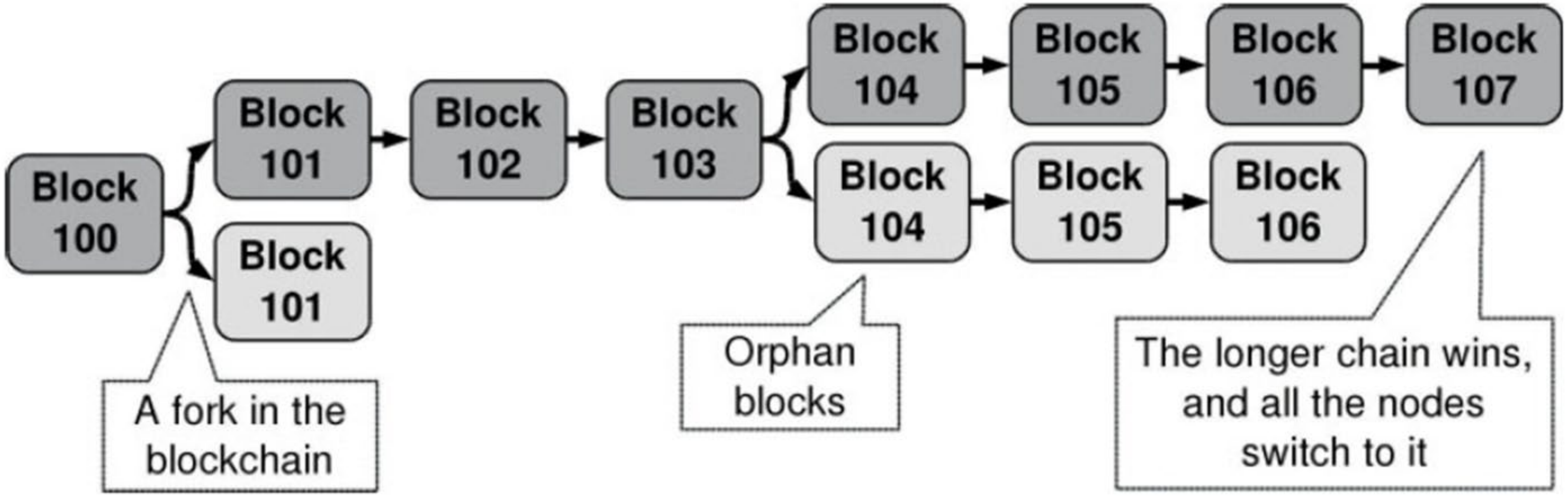


# Smartcontract or chaincode



```
note: *** An Ethereum smart contract to sell a website for "5000 by March"
note: First, store buyer's ethereum address:
put 6af26739b9ffef8aa2985252e5357fde in storage slot BUYER
note: Then, store seller's ethereum address:
put feab802c014588f08bfee2741086c375 in storage slot SELLER
note: April 1, 2014 is 1396310400 in "computer time"
put 1396310400 in storage slot DEADLINE
note: If the agreed amount is received on time...
when
        transaction value ≥ 5000 ether
    and
        block timestamp ≤ storage slot DEADLINE
then
    note: ... then designate the buyer as the new website admin and pay the seller
    put storage slot BUYER in storage slot WEBSITE_ADMIN
    spend contract balance to storage slot SELLER
```

# Forking


Block 100
Block 101
Block 101
Block 102
Block 103
Block 104
Block 105
Block 106
Block 107
Block 104
Block 105
Block 106
A fork in the blockchain
Orphan blocks
The longer chain wins, and all the nodes switch to it

# CIA … what are the results of the research project ?

BLOCKCHAIN4SME

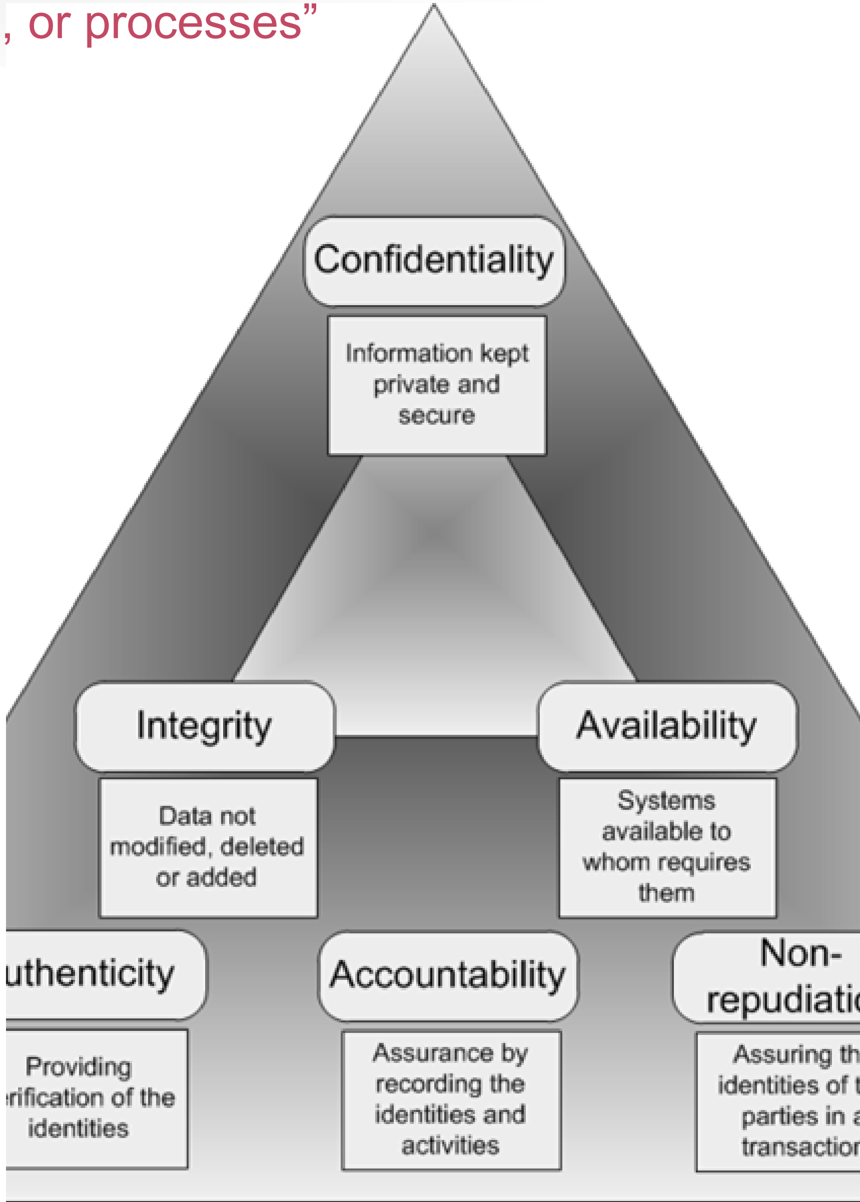
# Confidentiality

According to the National Institute of Standards and Technology (NIST), confidentiality refers to “the property that sensitive information is not disclosed to unauthorized individuals, entities, or processes”

- **Network access : firewall , VPN , VLAN , IDS , …**
- **Access control on application level**
- **Information Security Management System**

- **Cryptography : key generation**
- **PKI : Public Key Infrastructure**
- **Full encryption of the data blocks** →Authentication & authorization controls

- **Key management** →key storage , key loss , key theft
- **Wallet management** → Key theft , unauthorised access to data
- **Quantum resistant cryptography** → SHA-256 replaced by SHA-384



Confidentiality
Information kept private and secure
Integrity
Data not modified, deleted or added
Availability
Systems available to whom requires them
uthenticity
Providing rification of the identities
Accountability
Assurance by recording the identities and activities
Non-repudiati
Assuring th identities of t parties in a transactio

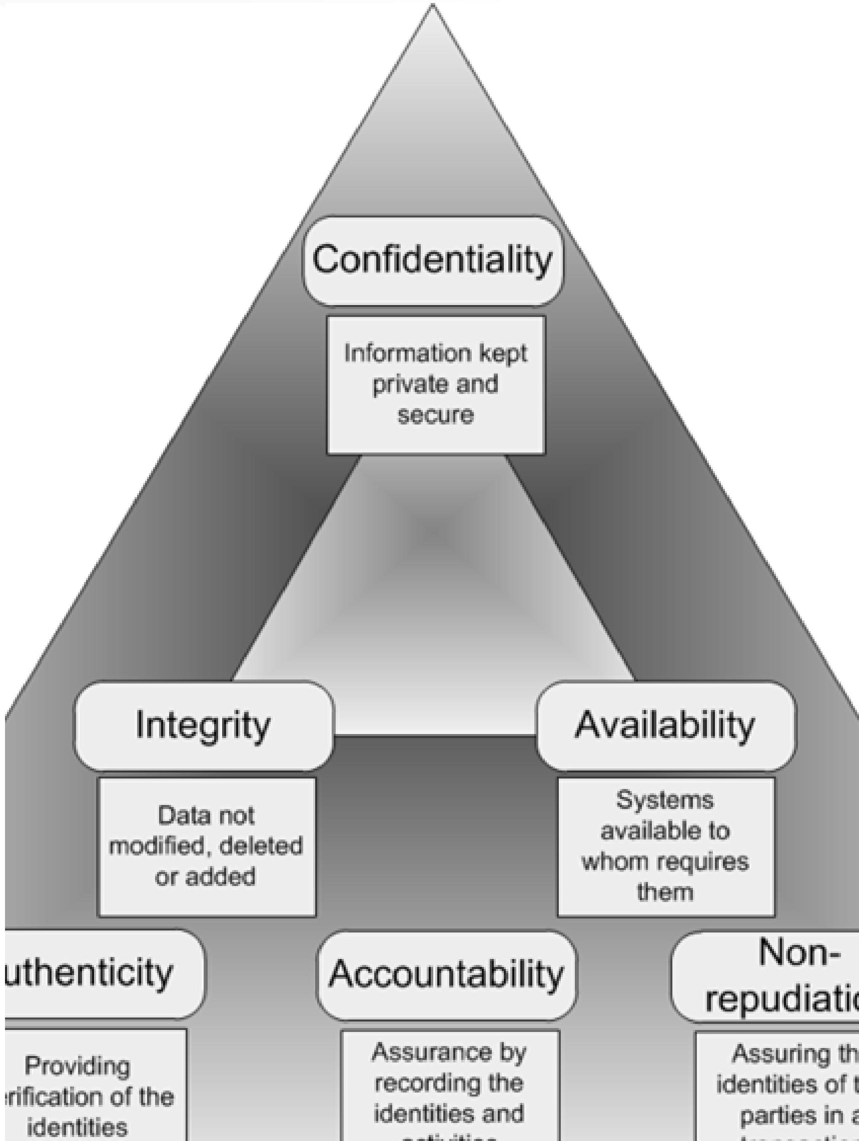
# Integrity

Integrity is defined as the "guarding against improper information modification or destruction, and includes ensuring information non-repudiation and authenticity"according to NIST

- **Data encryption - hash comparison – digital signing**

- **Immutability ->** sequential hashing and cryptography + distributed
- **Consensus models**
- **Tracebility – non repudiation ->** time stamped and digital signed

- **Smart contracts** → S-SDLC
- **Data quality** → Trusted oracles : data feed third party service in smart contracts
- **GDPR** → Right to be forgotten
- **Consensus Hijack** → Fraudulent transactions - Sybil attaque



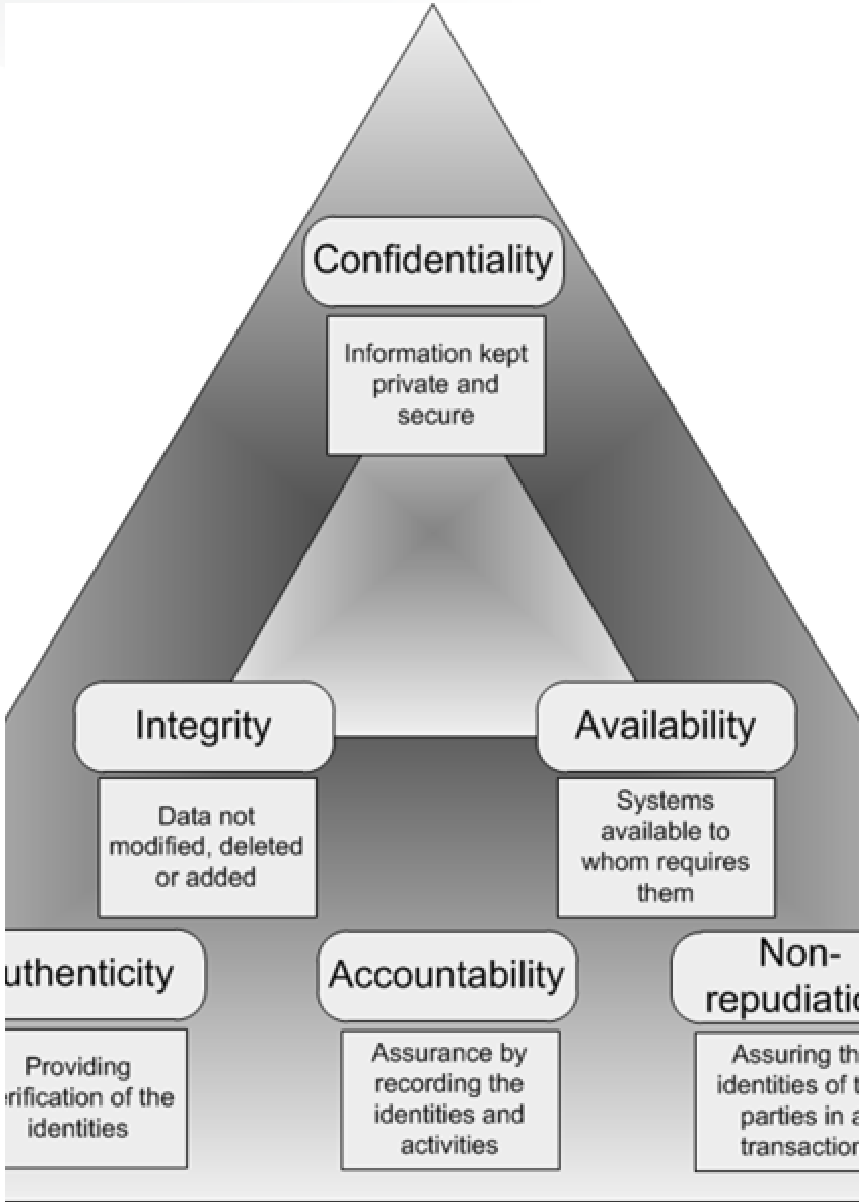
Confidentiality
Information kept private and secure
Integrity
Data not modified, deleted or added
Availability
Systems available to whom requires them
uthenticity
Providing rification of the identities
Accountability
Assurance by recording the identities and
Non-repudiatio
Assuring th identities of parties in a

# Availability

NIST defines availability as "ensuring timely and reliable access to and use of information"

- **No single point of failure →** IP based DDos no effect
- **Operational Resilience** → Distributed nodes , peer to peer, 24/7

- **Global internet outage**
- **Scalability →** unexpected growth of the DLT database
- **Denial of Service** → large volumes of small transactions


Confidentiality
Information kept private and secure
Integrity
Data not modified, deleted or added
Availability
Systems available to whom requires them
uthenticity
Providing rification of the identities
Accountability
Assurance by recording the identities and activities
Non-repudiati
Assuring th identities of t parties in a transactio

# ISO/TC 307
# Blockchain and distributed ledger technologies

# Existing Threats

- The first happens at the level of the transaction itself. In this category, the source of the threat is the behavior of a user, because of the user's incompetence or dishonesty. One example of this category is a double-spending attack.
- The second happens at the level of transaction validation. In this category, the threat comes from the  collective behavior of dishonest miners. One example in this category is the 51% attack problem.

# Existing vulnerabilities

- User layer vulnerabilities

- **User apps vulnerabilities**
- **Admin apps vulnerabilities**

# Existing vulnerabilities

- API layer vulnerabilities

- **External interfaces vulnerabilities**
- **User API vulnerabilities**
- **Admin API vulnerabilities**

# Existing vulnerabilities

- Platform layer vulnerabilities
- **Consensus mechanism vulnerabilities**

  During Finney attack

  Brute force attack

  The race attack

  Vector 76 or one-confirmation attack

  Punitive and Feather Forking

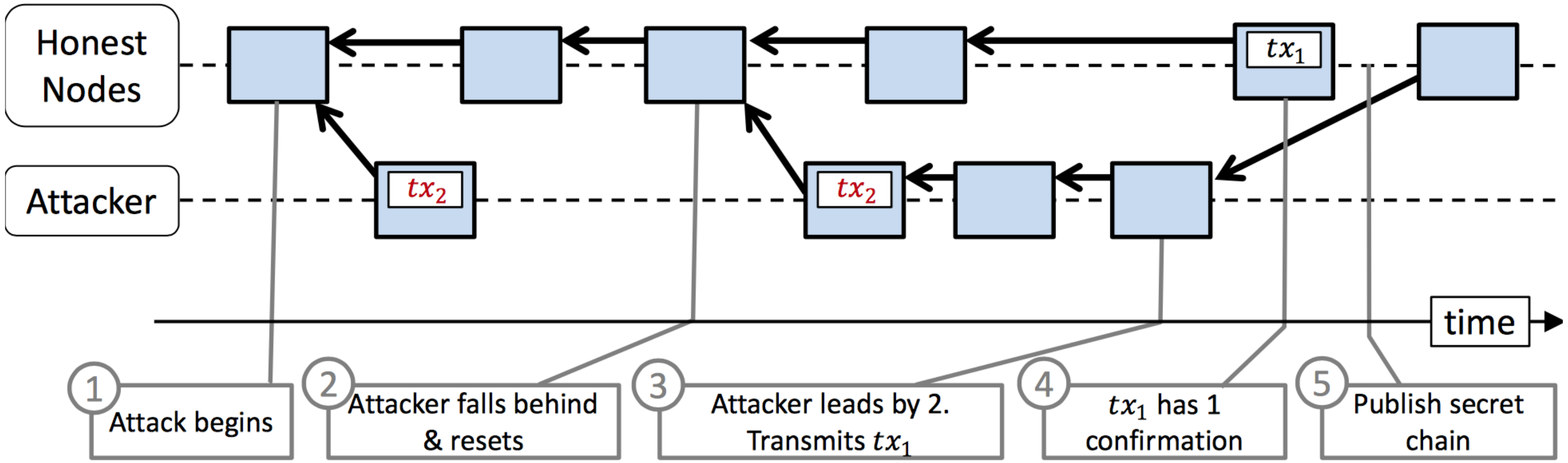
  Goldfinger attack

  Selfish mining attack

  Coin-hopping attack

  Nothing at stake attack

# Vector 76 or one-confirmation attack is a combination of the race attack and the Finney attack


Honest Nodes
Attacker
$tx_1$
$tx_2$
$tx_2$
time
1 Attack begins
2 Attacker falls behind & resets
3 Attacker leads by 2. Transmits $tx_1$
4 $tx_1$ has 1 confirmation
5 Publish secret chain

*As the attack begins the attacker starts working on a secret chain with $tx_2$ inside its first block (1). If the attacker's chain is shorter than the honest nodes', the attacker gives up and restarts the attack (2). The attacker manages to gain a lead of 2 blocks (3). He then transmits the transaction he wishes to double spend which is included in a block (4). The transaction now has enough confirmations (1-conf) and the attacker collects his rewards. He then publishes his secret chain and successfully double spends (5). Notice that once the pre-mining stage is concluded, the attack succeeds with probability 1, so miners that see $tx_1$ that is only broadcast then will always lose the funds.*

# Existing vulnerabilities

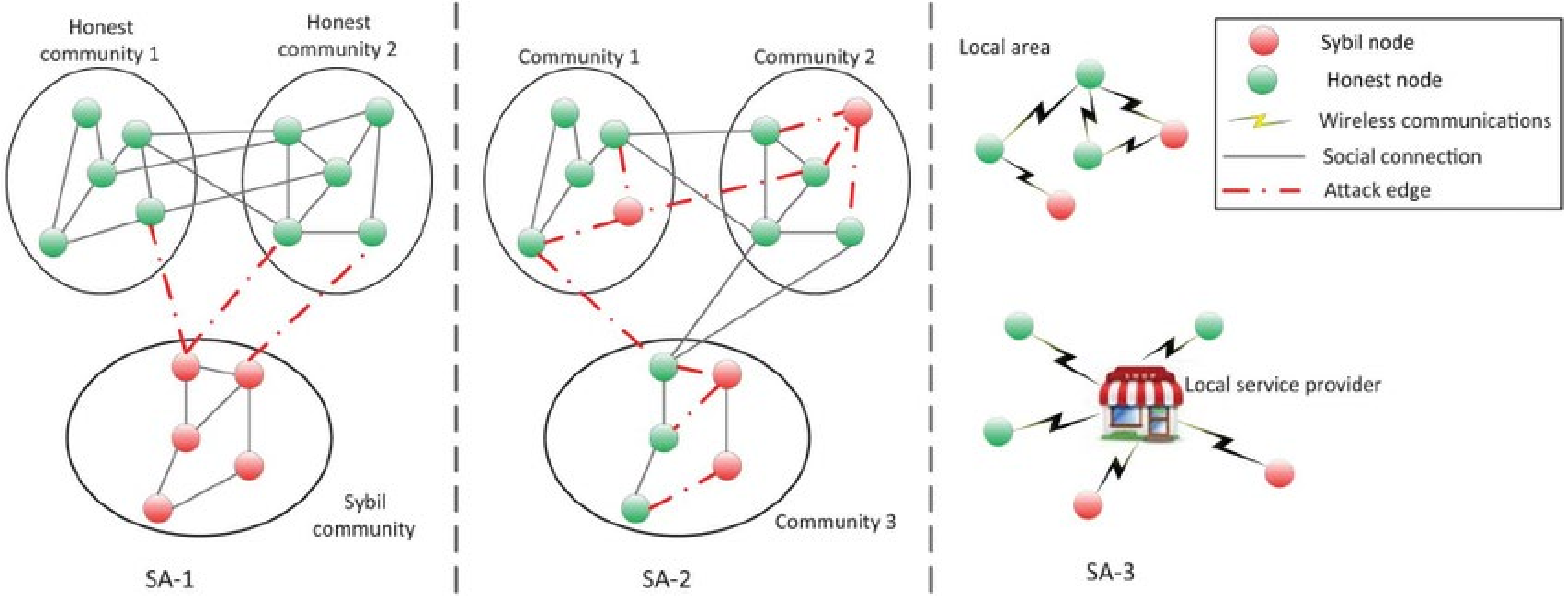
- Platform layer vulnerabilities

- Membership services vulnerabilities

  Sybil attack (multiple virtual identities)

- Event distribution vulnerabilities
- Crypto services vulnerabilities
- State management vulnerabilities
- Smart contract vulnerabilities

# Sybil attack (multiple virtual identities)



# Existing vulnerabilities

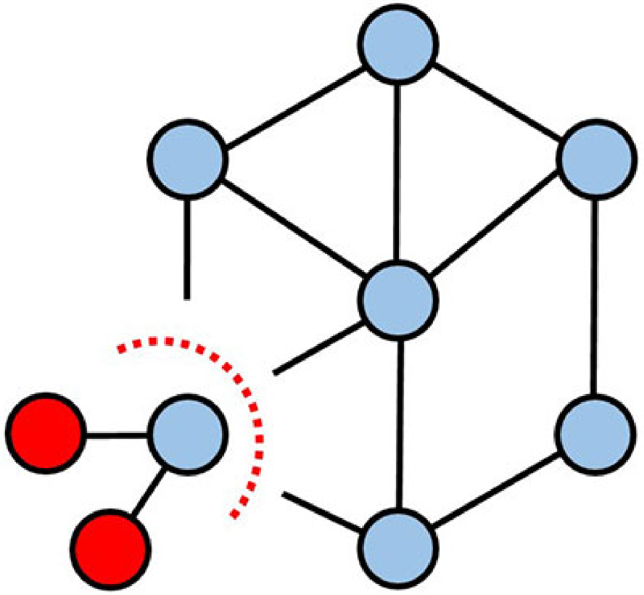
- Infrastructure layer vulnerabilities
- Storage vulnerabilities
- P2P network vulnerabilities

  Eclipse or net-split attack

  Tampering an adversary

- Runtime environment vulnerabilities
- Vulnerability used by implementation flaw

# Eclipse or net-split attack



# Existing risks

- Disclosure of private information and cryptographic keys
- Denial of service
- Forking in blockchain and DLT
- Compromise of Cryptography
- Data poisoning

CONCLUSION

# Apply What You Have Learned Today

- New technologies bring new risks with them
- To do : Blockchain Preparation Audit Program

1. Pre-implementation
2. Governance
3. Development
4. **Security**
5. Transactions
6. Consensus

# Blockchain Preparation Audit Program : Security

## Wallet Management

- Private keys are secured appropriately.
- The enterprise has implemented a process for managing loss or theft of private keys.

## Secure Coding

- Source code repositories are secure.
- Source code is reviewed for vulnerabilities.
- Vulnerabilities identified during source-code reviews are properly managed in terms of mitigation, action plans and communication to relevant stakeholders.

# Blockchain Preparation Audit Program : Security

### Network-Vulnerability Management

- A process is in place to manage blockchain network vulnerabilities.
- The process for managing blockchain network vulnerabilities is operationally effective and demonstrable.

### Endpoint Security

- A process exists to manage endpoint security for devices using the blockchain solution.
- The process for managing endpoint security is operationally effective and demonstrable.

# Private keys are secured appropriately.

- Ensure that private keys are appropriately secured. Consider the following:

a. Use of software (client side or online) vs. hardware wallets

b. Use of hot (live) vs. cold storage (offline or airgap)

c. Use of multifactor authentication

d. Use of password to encrypt local storage

e. Backup of private keys or 12-word phrase (i.e., master seed)

f. Segregation of backup from primary use point

- Ensure that the enterprise has a policy for securing private keys that has been approved by the relevant stakeholders.

- Review logical access to determine whether appropriate personnel manage private keys; ensure that there is an adequate segregation of duties."

# The enterprise has implemented a process for managing loss or theft of private keys.

- Verify that the enterprise has an adequate insurance policy. Determine whether:

  a. Appropriate financial and reputational protection exists for the enterprise and its clients.

  b. Adequate subject matter experts have been consulted for input (e.g., experts in risk, legal and information security).

- Verify that the enterprise communicates loss of private keys appropriately.

a. Determine whether the enterprise has a process to notify appropriate parties—both internal (e.g., senior management) and external (e.g., clients and regulators)—in the event that private keys are lost or stolen.

b. Verify that the process is consistent with the enterprise's incident-communication strategy and consistent with a response in the event of theft of sensitive customer data.

# Source code repositories are secure.

- For permissionless repositories (e.g., GitHub), ensure that security is reasonable. Consider the following:

a. Reputation of repository (including known security incidents)

b. Process for approving source-code changes (including input from core developer group, community feedback, approval of changes)

c. Activities of the repository and degree of community engagement (e.g., number of active contributors, number of commits, pull requests, active issues, etc.)

- For permissioned repositories (e.g., private or consortium), ensure that adequate security controls exist. Verify that:

a. Appropriate security controls are in place for code repositories (e.g., segregation of duties, approval process for changes, access controls).

b. Policies and procedures are documented and understood by all parties, where code repositories are shared by the enterprise via consortium.

# Source code is reviewed for vulnerabilities.

- Ensure that adequate code reviews take place. Verify the following:

a. For permissionless blockchains, source code is vetted at least quarterly though manual code review, penetration tests and/or automated scans.

b. For permissioned blockchains, source code is reviewed in accordance with relevant policies and procedures.

c. Source code is independently reviewed by qualified security professionals with experience in the enterprise's specific blockchain platform(s).

- Determine whether appropriate stakeholders participate in the code review process (e.g., information security, information technology stakeholders).

# Vulnerabilities identified during source-code reviews are properly managed in terms of mitigation, action plans and communication to relevant stakeholders.

- Verify that an adequate remediation process is in place for identified source-code vulnerabilities. Determine whether:

a. For permissionless blockchains, the enterprise has considered appropriate actions (e.g., forking to a different blockchain, limiting certain transactions).

b. For permissioned blockchains, the enterprise has considered actions consistent with relevant policies and procedures.

- Verify that the process for remediating blockchain source-code vulnerabilities has been approved by relevant stakeholders.

- Select a sample of identified blockchain source-code vulnerabilities and verify adherence to the blockchain source-code remediation process.

# A process is in place to manage blockchain network vulnerabilities

- Review the blockchain network-vulnerability management process for adequacy. Determine whether the following provisions exist:

  a. Monitoring for blockchain vulnerabilities (e.g., 51% attack, double-spend attack, malicious smart contracts, denial-of-service (DoS) attack, Sybil attack, packet sniffing)

  b. Periodic execution of automated vulnerability-assessment solution

  c. Remediation protocol for identified blockchain vulnerabilities (e.g., forking, halting transactions)

  d. Escalation protocol for identified vulnerabilities and a plan for communication to relevant stakeholders

- Verify that the blockchain network-vulnerability management process has been approved by relevant stakeholders.