The Blockchain: What It is & Why It Matters

Abhishek Dubey
Anastasia Mavridou
Douglas C. Schmidt

This work has been funded in part by Siemens, Varian, & Accenture
Overview of the Presentation

- Provide an introduction to the blockchain & why it matters to the community
Overview of the Presentation

• Provide an introduction to the blockchain & why it matters to the middleware IOT community
• Explore challenges to applying blockchain for various domains, including IOT & beyond
What is a Blockchain?
What is a Blockchain?

A blockchain is a decentralized platform that supports “trustless” transactions.

See en.wikipedia.org/wiki/Blockchain
What is a Blockchain?

• A blockchain is a decentralized platform that supports “trustless” transactions

Used as computational substrate for “crypto-currencies”, plus more
What is a Blockchain?

- A blockchain is a decentralized platform that supports “trustless” transactions

A cryptocurrency is a digital asset that uses cryptography to secure transactions, control the creation of additional units, & verify asset transfer

Used as computational substrate for “cryptocurrencies”, plus more
What is a Blockchain?

- A **blockchain** is a decentralized platform that supports “trustless” transactions.
What is a Blockchain?

- A blockchain is a decentralized platform that supports “trustless” transactions.

This chain of blocks provides an open, distributed ledger that immutably records transactions between two parties efficiently & verifiably.
What is a Blockchain?

- A **blockchain** is a decentralized platform that supports “trustless” transactions

A block contains transaction data, a timestamp, & a hash pointer that links to the previous block (which forms the “chain” of blocks)
What is a Blockchain?

- A blockchain is a decentralized platform that supports “trustless” transactions.

These blocks are replicated across many computers, rather than being stored on a central server.
What is a Blockchain?

- A blockchain is a **decentralized platform** that supports “trustless” transactions.

This platform may be distributed globally (public blockchain).
What is a Blockchain?

• A blockchain is a decentralized platform that supports “trustless” transactions.

Public blockchains are less efficient, but most useful when not all participants can be trusted to behave.
What is a Blockchain?

- A blockchain is a **decentralized platform** that supports “trustless” transactions.

  It could be localized to a limited group (private blockchain).
What is a Blockchain?

- A blockchain is a **decentralized platform** that supports “trustless” transactions.

Each node in a blockchain network runs common middleware & all nodes have equal level of privilege & access.
What is a Blockchain?

- A blockchain is a decentralized platform that supports “trustless” transactions.

Blockchain middleware provides services to applications beyond what's provided by the OS & communication protocols.
What is a Blockchain?

A blockchain is a decentralized platform that supports “trustless” transactions. Blockchain middleware enables anonymous exchange of digital assets without the need for a central authority to verify trust & transfer of value.
What is a Blockchain?

A blockchain is a decentralized platform that supports "trustless" transactions.

Key characteristics of a blockchain:
- Blockchain protocol
- Immutable replicated database
- Identity, reputation

On top of:

Traditional Internet:
- Store & copy

Digital Asset Transaction:
- Record & transfer

Institute for Software Integrated Systems
World-class, interdisciplinary research with global impact.
What is a Blockchain?

- A blockchain is a decentralized platform that supports "trustless" transactions.

Ensures a common, unambiguous ordering of blocks & guarantees the (eventual) integrity & consistency of the blockchain across (geographically) distributed nodes.
What is a Blockchain?

- A blockchain is a decentralized platform that supports "trustless" transactions

A database containing immutable time-stamped information for every transaction that’s replicated on servers (may be around the world)
What is a Blockchain?

- A blockchain is a decentralized platform that supports "trustless" transactions. It's extremely hard to change a blockchain without collusion, and it's extremely easy to detect the attempt if anyone tries.
What is a Blockchain?

- A blockchain is a decentralized platform that supports "trustless" transactions.

Build a sense of trust among anonymous users of online communities.
Why Blockchain Matters
Why Blockchain Matters

• Blockchain helps address needs in various domains by providing decentralized “transactions-as-a-service”
Why Blockchain Matters

• Blockchain helps address needs in various domains by providing decentralized “transactions-as-a-service”

A key goal is to “disintermediate” centralized brokers, yet still allow multiple parties (who don’t trust each other) to share a single database.
Why Blockchain Matters

- Blockchain helps address needs in various domains by providing decentralized “transactions-as-a-service”, e.g.
  - Payments in the financial sector
    - e.g., use on-chain tokens to represent cash, stocks, bonds, etc

Turns out to be problematic in practice due to lack of confidentiality.
Why Blockchain Matters

• Blockchain helps address needs in various domains by providing decentralized “transactions-as-a-service”, e.g.
  • Payments in the financial sector
  • Lightweight financial systems, e.g.
  • Streamline interactions between commuters & multi-modal transit

Why Blockchain Matters

- Blockchain helps address needs in various domains by providing decentralized “transactions-as-a-service”, e.g.
  - Payments in the financial sector
  - Lightweight financial systems, e.g.
    - Streamline interactions between commuters & multi-modal transit
  - Improve efficiency of energy transactions in “smart grids”

Why Blockchain Matters

• Blockchain helps address needs in various domains by providing decentralized “transactions-as-a-service”, e.g.
  • Payments in the financial sector
  • Lightweight financial systems, e.g.
    • Streamline interactions between commuters & multi-modal transit
    • Improve efficiency of energy transactions in “smart grids”
  • UN provides thousands of Syrian refugees in Jordan with food, clothing, & other aid in a cost effective manner

Why Blockchain Matters

- Blockchain helps address needs in various domains by providing decentralized “transactions-as-a-service”, e.g.
  - Payments in the financial sector
  - Lightweight financial systems
  - Interorganizational record keeping
  - e.g., provide providers, patients, (& surrogates) better access to —& control over—health info

See www.dre.vanderbilt.edu/~schmidt/PDF/PLoP-2017-blockchain.pdf
Blockchains and Cyber-Physical Systems (CPS)
Blockchains Increasingly are Being Used in CPS

- The notion of decentralized computation & trustless computing also provides opportunities/challenges in the IoT
The Reason is the Focus on Decentralization
Key Requirements For Decentralized CPS

Decentralized Control

- Compute control actions using distributed averaging consensus within a specific time limit
- Requires real-time information dissemination and time synchronized task execution
Key Requirements For Decentralized CPS

Decentralized Information

- Preserve integrity of Information across all actors in the system
- Support for information aggregation and transactions
- Requires consensus, and distributed ledger

Microgrids

Transportation

Health Care

Smart Buildings
Transactive Energy Systems: An Example
Example Application: Private And Decentralized Energy Transactions

Application use case for integration of RIAPS and Blockchains
Sequence Diagram

DSO | Prosumer | Consumer | Smart Contract/Blockchain | Solver | Event Recorder

Connect | Connect | Connect | Connect | Connect

query_contract_address | query_contract_address | query_contract_address | query_contract_address

Contract Address | Contract Address | Contract Address

register | register | register | register

postBuyingOffer | postBuyingOffer | postBuyingOffer

BuyingOfferPosted | BuyingOfferPosted | BuyingOfferPosted

SellingOfferPosted | SellingOfferPosted | SellingOfferPosted

Solve

submitSolution
Evaluation

- We used real-world energy production / consumption data from a German microgrid provided by Siemens, CT.
- We deployed our system on a private Ethereum network.
- 5 producers
- ~97 consumers
Enabling safe & private interactions with blockchains in smart grid
(see arxiv.org/abs/1709.09597?context=cs.DC)
There are still Challenges
Ethereum Smart Contracts
Ethereum Smart Contracts

- Smart contracts are programs that run on the blockchain

```solidity
contract MyToken {
    /* This creates an array with all balances */
    mapping (address => uint256) public balanceOf;
    /* Initializes contract with initial supply tokens to the creator of the contract */
    function MyToken(uint256 initialSupply) {
        balanceOf[msg.sender] = initialSupply; // Give the creator all initial tokens
    }
    /* Send coins */
    function transfer(address _to, uint256 _value) {
        if (balanceOf[msg.sender] < _value) throw; // Check if the sender has enough
        if (balanceOf[_to] + _value < balanceOf[_to]) throw; // Check for overflows
        balanceOf[msg.sender] -= _value; // Subtract from the sender
        balanceOf[_to] += _value; // Add the same to the recipient
    }
}
```
Security Vulnerabilities

- Contracts are riddled with bugs and security vulnerabilities
- A recent automated analysis of 19,336 contracts Ethereum contracts
  - 8,333 contracts suffer from at least one security issue

Why is that Important?

- Smart Contracts handle financial assets of significant value!
- The value held by Ethereum contracts is: **12,205,760 Ethers**
  - This is around **$11 Billion**
Why is that Important?

- Smart contract bugs cannot be patched
  - Once a contract is deployed, its functionality cannot be altered

- Blockchain transactions cannot be rolled back
  - Once a malicious transaction is recorded it cannot be removed
  - “Code is law” principle

Institute for Software Integrated Systems
World-class, interdisciplinary research with global impact.
A Transaction can be Rolled Back..

- with a **hard fork** of the blockchain
  - Requires consensus among all stakeholders
  - Undermines the trustworthiness of the platform
  - Ethereum forked the blockchain to undo the Dao attack
The Infamous DAO Attack

- The DAO was a contract with ~$150M built by Ethereum creators
  - A combination of vulnerabilities was exploited
  - Attackers stole 3.6M Ethers, worth ~$60M at the time of the attack
  - Re-entrancy vulnerability
Re-entrancy Vulnerability

• In Ethereum, when there is a function call
  • The caller has to wait for the call to finish - synchronous calls
  • A malicious callee might take advantage of this

```
function withdraw(uint amount) {
  if (credit[msg.sender] >= amount) {
    msg.sender.call.value(amount)();
    credit[msg.sender] -= amount;
  }
}
```

DAO...
Re-entrancy Vulnerability

• In Ethereum, when there is a function call
  • The caller has to wait for the call to finish - synchronous calls
  • A malicious callee might take advantage of this

```solidity
function withdraw(uint amount) {
    if (credit[msg.sender] >= amount) {
        msg.sender.call.value(amount)();
        credit[msg.sender] -= amount;
    }
}
```

```solidity
function() {
    dao.withdraw(dao.queryCredit(this));
}
```
Re-entrancy Vulnerability

- In Ethereum, when there is a function call
  - The caller has to wait for the call to finish - synchronous calls
  - A malicious callee might take advantage of this

```solidity
function withdraw(uint amount) {
    if (credit[msg.sender] >= amount) {
        msg.sender.call.value(amount)();
        credit[msg.sender] -= amount;
    }
}
```

```solidity
function () {
    dao.withdraw(dao.queryCredit(this));
}
```
Re-entrancy Vulnerability

• In Ethereum, when there is a function call
  • The caller has to wait for the call to finish - synchronous calls
  • A malicious callee might take advantage of this

```solidity
function withdraw(uint amount) {
    if (credit[msg.sender]>= amount) {
        msg.sender.call.value(amount)();
        credit[msg.sender] -= amount;
    }
}
```

```solidity
function() {
    dao.withdraw(dao.queryCredit(this));
}
```
Re-entrancy Vulnerability

- In Ethereum, when there is a function call
  - The caller has to wait for the call to finish - synchronous calls
  - A malicious callee might take advantage of this

```solidity
function withdraw(uint amount) {
    if (credit[msg.sender] >= amount) {
        msg.sender.call.value(amount)();
        credit[msg.sender] -= amount;
    }
}
```

```solidity
function() {
    dao.withdraw(dao.queryCredit(this));
}
```
Unpredictable State Vulnerability

• The order of execution of function calls cannot be predicted
• No prior knowledge of a contract’s state during call execution
Unpredictable State Vulnerability

- The order of execution of function calls cannot be predicted
- No prior knowledge of a contract’s state during call execution

createOffer (sell 10 tokens for 1 ether)

Seller → Market → Buyer

createOffer (sell 10 tokens for 1 ether)

Institute for Software Integrated Systems
World-class, interdisciplinary research with global impact.
Unpredictable State Vulnerability

- The order of execution of function calls cannot be predicted
- No prior knowledge of a contract’s state during call execution

Game-Theoretic Model

- for each user $u$, selects a false negative rate $f_u$
- we assume that the feasible FP / FN rate pairs are given by a function $FP(f_u)$
- selects a set of users $A$, and sends them targeted malicious e-mails
- can select at most $A$ users (otherwise the attack is easily detected)

Defender Targeting attacker $f_u$ FP Non-targeting attacker(s) non-strategic (not a player)

createOffer (sell 10 tokens for 1 ether)

Market

acceptOffer

Seller

Buyer
Unpredictable State Vulnerability

• The order of execution of function calls cannot be predicted
• No prior knowledge of a contract’s state during call execution

Game-Theoretic Model

- for each user $u$, selects a false negative rate $f_u$
- we assume that the feasible FP / FN rate pairs are given by a function $FP(f_u)$
- selects a set of users $A$ and sends them targeted malicious e-mails
- can select at most $|A|$ users (otherwise the attack is easily detected)

Defender

- targeting attacker $f_u$
- non-strategic (not a player) ($\Phi$)

Buyer

createOffer (sell 10 tokens for 1 ether)

updateOffer (sell 1 token for 1 ether)

Market

acceptOffer

createOffer (sell 10 tokens for 1 ether)

updateOffer (sell 1 token for 1 ether)

Institute for Software Integrated Systems
World-class, interdisciplinary research with global impact.
The Reason behind many Vulnerabilities

• Vulnerabilities often arise due to the semantic gap between
  • The underlying execution semantics
  • The actual semantics of smart contracts
• There exist tools for identifying common vulnerabilities in existing contracts
• We explore a different avenue
  • We want to help developers to create secure and correct smart contracts
Our Approach

- Relies on the following observations. Smart contracts:
  - Have states
  - Provide functions that can be invoked and change the contract state
- Smart contracts can be naturally represented by state machines
- Adequate level of abstraction for reasoning about their behavior

State Machines

- Taking a transition
  - Is allowed if the guard evaluates to true
  - Executes the action
  - Updates the contract’s current state

**Definition 1.** A Smart Contract is a tuple \((S, s_0, C, I, O, \rightarrow)\), where:
- \(S\) is a finite set of states;
- \(s_0 \in S\) is the initial state;
- \(C, I,\) and \(O\) are disjoint finite sets of, respectively, contract, input, and output variables;
- \(\rightarrow \subseteq S \times G \times \mathcal{F} \times S\) is a transition relation, where:
  - \(G = \mathcal{B}[C, I]\) is a set of guards;
  - \(\mathcal{F}\) is a set of action sets, i.e., a set of all ordered powersets of \(\mathcal{E}[C, I, O]\)
FSolidM Demo
Examples of FSolidM Plugins

• Locking plugin

```solidity
bool private locked = false;
modifier locking {
    require(!locked);
    locked = true;
    _;
    locked = false;
}
```

• Reentrancy vulnerability

• Transition Counter

```solidity
uint private transitionCounter = 0;
modifier transitionCounting(uint nextTransitionNumber) {
    require(nextTransitionNumber == transitionCounter);
    transitionCounter += 1;
    _;
}
```

• Unpredictable state vulnerability
Current Work on FSolidM

FSM graphical editor

1

Solidity code editor

2

Verification

3

Solidity code generation

4

Secure smart contract

Tool and publications available online:

http://cps-vo.org/group/SmartContracts

Source code:

http://github.com/anmavrid/smart-contracts
Verification of Smart Contracts

- Deadlock-freedom analysis
  - Parity wallet vulnerability was based on a deadlocked contract
- Functional property analysis

<table>
<thead>
<tr>
<th>Transition</th>
<th>Statement</th>
<th>Liveness Not verified</th>
<th>Safety Verified</th>
<th>Safety Verified</th>
</tr>
</thead>
<tbody>
<tr>
<td>AG(\text{call} \rightarrow \text{AX} A[\neg \text{call} \lor \text{subtract}])</td>
<td>if call happens, call can only happen after subtract</td>
<td>DAO attack</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The FSolidM Framework

- Formal model, clear semantics easy-to-use graphical editor
  - Decreasing the semantic gap
- Rigorous semantics
  - Amenable to analysis and verification
- Code generation + functionality and security plugins
  - Minimal amount of error-prone manual coding
- Tool and publications: [http://cps-vo.org/group/SmartContracts](http://cps-vo.org/group/SmartContracts)
- Source code: [http://github.com/anmavrid/smart-contracts](http://github.com/anmavrid/smart-contracts)

The FSolidM Framework

- Formal model, clear semantics easy-to-use graphical editor
  - Decreasing the semantic gap
- Rigorous semantics
  - Amenable to analysis and verification
- Code generation + functionality and security plugins
  - Minimal amount of error-prone manual coding
- Tool and publications: [http://cps-vo.org/group/SmartContracts](http://cps-vo.org/group/SmartContracts)
- Source code: [http://github.com/anmavrid/smart-contracts](http://github.com/anmavrid/smart-contracts)