How Formal Analysis and Verification Add Security to Blockchain-based Systems

January 26, 2017
Shin’ichiro Matsuo (MIT Media Lab)
Pindar Wong (VeriFi Ltd.)
Outline of this talk

Security Definition of Blockchain-based system

Technology and Security Layer

Applicability of Formal Analysis and Verification

Four layers are suitable: Implementation, Backbone Protocols, Application Protocols and Application Logic

Idea toward Domain specific language
Background: The case of “the DAO”

Had chance to lose 50M Dollars by this attack.

Caused by vulnerability of the code

The way of workaround is still not decided.

Problems

Vulnerability handling

Procedure for work around

Over-investment to uncertified technology and codes

Intersection of technology and financial incentive
Security definitions of blockchain

Several proposals on back-bone protocol

Few consideration for security of entire system
Security Definitions for backbone-protocol [1]

Two definitions

Common Prefix Property
If two players prune a sufficient number of blocks from their chains, they will obtain the same prefix.

Chain Quality
Any large enough chunk of an honest player’s chain will contain some block from honest players.

There are results on provable secure protocol but needs assumptions [KKRDO16]
Provable Secure Blockchain with Proof of Stake [KKRDO16]

Prove Two Requirements of Blockchain

Persistence and Liveliness [1]: Robustness of the Blockchain

Propose Provable Secure Protocol

Use Multi-Party Coin Flipping for leader election to produce randomness

Many Assumptions

Highly Synchronous
Majority of Selected Stakeholder is available
The Stakeholders do not remain offline for a long time
How Formal Analysis and Verification Add Security to Blockchain-based Systems

Example of Blockchain Technology layer [3]

- **Network**: broadcasting transactions and blocks
- **Consensus**: the agreement-reaching engine
- **Storage**: bootstrapping new nodes, storing archival data
- **Application**: transaction graph, scripting language semantics
- **View**: cached summary of the transaction log
- **Side-plane**: off-chain contracts
Layers for security consideration

- **Operation**: Key Management, Audit, Backup (ISO/IEC 27000)
- **Implementation**: Program Code, Secure Hardware (ISO/IEC 15408)
- **Application Logic**: Scripting Language for Financial Transaction, Contract (Secure coding guides)
- **Application Protocol**: Privacy protection, Secure transaction (ISO/IEC 29128)
- **Backbone Protocol**: P2P, Consensus, Merkle Tree (ISO/IEC 29128)
- **Cryptography**: ECDSA, SHA-2, RIPEMD160 (NIST, ISO)
Cryptography Layer

Security goals in Blockchain
Realizing authenticity and integrity

Digital Signature: ECDSA
Hash Function: SHA-2, RIPEMD-160
Underlying Mathematics: Secure parameter of elliptic curve

Firm analysis model
Provable Security
Estimation of security margin

Many theoretic results and evaluations

Academic proof, Standardization by NIST, ISO/IEC, IETF(IRTF), IEEE
Backbone Protocol Layer

Security goals in Blockchain

Realizing de-centralization and robustness by P2P network
Realizing consistency of transaction by consensus algorithm
Ensuring order of transaction by Merkle hash tree and chaining

Security definition, requirements and evaluation

No fixed security definition (researches are ongoing)
Evaluation by mathematical proof or formal analysis

Standard for evaluation

ISO/IEC 29128 for cryptographic protocols
Application Protocol Layer

Security goals in Blockchain

Privacy Protection
Secure data transmission
Secure transaction

Security definition, requirements and evaluation

Need application specific security definition
Evaluation by mathematical proof or formal analysis

Standard for evaluation

ISO/IEC 29128 for cryptographic protocols
Application Logic Layer

Security goals in Blockchain

Soundness and completeness in application logic

Security definition, requirements and evaluation

Checking the existence of bug
Implementation Layer

Security goals in Blockchain

Protection of signing key and prevent forgery of digital signature

Against black box attacker (main channel), gray box attacker (Side channel) and white box attacker (rooted device)

Security definition, requirements and evaluation

Capability of the adversary

Standard for evaluation

ISO/IEC 15408
Operation Layer

Security goals in Blockchain

Key management
Audit of operation

Security definition, requirements and evaluation

Depends on security policy of each system

Standard for evaluation

ISO/IEC 27000 Series (Information Security Management System)
Formal Analysis and Formal Verification

Formal Analysis

Evaluating the possibility of attack on the specification of the protocol, products or system by conducting some mathematical formalization of the security requirements, specifications and operational environment (an adversarial model).

Formal Verification

To verify the correctness of the specification of the protocol, products or system formal methods such as automated axiomatic theorem proving or model checking.
### Applicability of formal verification

<table>
<thead>
<tr>
<th>Operation</th>
<th>Security Layer</th>
<th>Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key Management, Audit, Backup</td>
<td>ISO/IEC 27000</td>
<td></td>
</tr>
<tr>
<td>Implementation</td>
<td>Program Code, Secure Hardware</td>
<td>ISO/IEC 15408</td>
</tr>
<tr>
<td>Application Logic</td>
<td>Scripting Language for Financial Transaction, Contract</td>
<td>Secure coding guides</td>
</tr>
<tr>
<td>Application Protocol</td>
<td>Privacy protection, Secure transaction</td>
<td>ISO/IEC 29128</td>
</tr>
<tr>
<td>Backbone Protocol</td>
<td>P2P, Consensus, Merkle Tree</td>
<td>ISO/IEC 29128</td>
</tr>
<tr>
<td>Cryptography</td>
<td>ECDSA, SHA-2, RIPEMD160</td>
<td>NIST, ISO</td>
</tr>
</tbody>
</table>

### How Formal Analysis and Verification Add Security to Blockchain-based Systems

- **Cryptography**: ECDSA, SHA-2, RIPEMD160
- **Backbone Protocol**: P2P, Consensus, Merkle Tree
- **Application Protocol**: Privacy protection, Secure transaction
- **Application Logic**: Scripting Language for Financial Transaction, Contract
- **Implementation**: Program Code, Secure Hardware
- **Operation**: Key Management, Audit, Backup

**Standards**:
- ISO/IEC 27000
- ISO/IEC 15408
- ISO/IEC 29128
- NIST, ISO
Formal analysis methods and tools for cryptographic protocol

<table>
<thead>
<tr>
<th>Symbolic</th>
<th>Model checking</th>
<th>Theorem proving</th>
</tr>
</thead>
<tbody>
<tr>
<td>NRL</td>
<td>SCYTHEER</td>
<td>Isabelle/HOL</td>
</tr>
<tr>
<td>FDR</td>
<td>ProVerif</td>
<td></td>
</tr>
<tr>
<td>AVISPA</td>
<td>AVISPA (TA4SP)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Unbounded</td>
</tr>
<tr>
<td>Cryptographic</td>
<td>CryptoVerif</td>
<td>BPW (in Isabelle/HOL)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Game-based Security Proof (in Coq)</td>
</tr>
</tbody>
</table>
Formal analysis of Implementation

Both software/hardware implementation

Security mechanisms which use cryptographic algorithms, protocols, random number generator and key management mechanisms

Target of Evaluation

Crypto-token wallet (Hardware/Software)

HSM (Hardware Security Module)
Standards and Examples for Implementation Layer

**Industrial Standard**
**Common Criteria (ISO 15408)**
Define seven EALs (Evaluation Assurance Levels)

EAL6 requires semi formal analysis on the design and implementation
EAL7 requires fully formal analysis on design and implementation

**Examples of formal analysis for implementation**
**EAL6**

FeliCa IC chip RC-SA00
Crypto Library V1.0 on P60x080/052/040yVC(Y/Z/A)/yVG
Microcontrôleurs sécurisés SA23YR48/80B et SB23YR48/80B
Analysis of Cryptographic Protocols: Formal Verification vs UC Framework

Formal Verification

• Formal method
• Find the existence of insecure state
• Automated verification
• Tool-aided

Mathematical Proof

• Rigorous proof
• Estimate probability of attack
• Same as cryptographic Primitive
Formal Analysis of Cryptographic Protocols

• Check if the insecure state may happen in execution
  • Protocol specification
  • Adversarial model
  • Insecure states to be avoided

- Insecure states
- Protocol specification
- Adversarial Model

Formal Analysis tool

• Output if the insecure states may happen.
• If yes, output trace by which the insecure state is happen.
Formal Analysis of Backbone protocols and application protocols

Explore the existence of state against security goals (Security Properties)

Dolev-Yao Model

• Basically Cryptographic algorithm is idealized
• Only a party who has a decryption key obtains plaintext.
• The other party obtains nothing.
• Same treatment for digital signature and others
• An adversary can control communication channel.
UC Framework

- Define the ideal functionality, then prove that the actual protocol is indistinguishable against the ideal functionality.
Combination of Formal Analysis and Mathematical proof

- Combine the merit of formal analysis and mathematical rigorous proof.
- Many researches from 2002
  - Game-based evaluation
  - Cryptooverif
The case of SSL/TLS

Many attacks/vulnerabilities are found during this 5 years.

Heartbleed, Poodle, FREAK, DROWN, CCS Injection

Problems

No security proof

No procedure for verification of technology.

No experts on the verification of cryptographic protocols

Insufficient quality assurance of program code
The case of TLS 1.3 [6]

Academia

Formal Verification

IETF

TLS1.3 Specification

Developer/Open Source Community

TRON Conference

Add trust
# International Standard: ISO/IEC 29128

How Formal Analysis and Verification Add Security to Blockchain-based Systems

Shin'ichiro Matsuo

---

## Accuracy

<table>
<thead>
<tr>
<th>Protocol Assurance Level</th>
<th>PAL1</th>
<th>PAL2</th>
<th>PAL3</th>
<th>PAL4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Protocol Specification</td>
<td>PPS_SEMIFORMAL</td>
<td>PPS_FORMAL</td>
<td>PPS_MECHANIZED</td>
<td></td>
</tr>
<tr>
<td>Adversarial Model</td>
<td>PAM_INFORMAL</td>
<td>PAM_FORMAL</td>
<td>PAM_MECHANIZED</td>
<td></td>
</tr>
<tr>
<td>Security Property</td>
<td>PSP_INFORMAL</td>
<td>PSP_FORMAL</td>
<td>PSP_MECHANIZED</td>
<td></td>
</tr>
<tr>
<td>Self Assessment Evidence</td>
<td>PEV_ARGUMENT</td>
<td>PEV_HANDPROVEN</td>
<td>PEV_BOUNDED</td>
<td>PEV_UNBOUNDED</td>
</tr>
</tbody>
</table>

Protocol Assurance Levels defined in ISO/IEC 29128
Security consideration for Smart contract

Need completeness and soundness as an application logic

The DAO case was caused by bug

Checking program code is well-known application of formal analysis
Language for Smart Contract

Solidity
Flexible and General purpose language

Bhargavan et al. proposed a framework to analyze both the runtime safety and functional correctness of a Solidity contract [9]

Introducing intermediate functional programming language suitable for verification

At this time, not covered all EVM functionalities
Designing Domain Specific Language

To limit possible execution states, which include “insecure” state, create a new domain-specific language.

Has enough capability to write business logic.

Suitable for formal verification.
Letter of Credit (L/C) and Trade Finance over Blockchain

How Formal Analysis and Verification Add Security to Blockchain-based Systems

Shin’ichiro Matsuo
Sequence of process

Contract

A,B

L/C

A

C

Payment

Issue Request

Lock Payment Value

Read

Read

Blockchain

Write

Settlement

Completion of trade

C,D

A

B

C

A
State Transitions of common process of L/C
Four variables for state representation: Contract, L/C, Payment, Shipment
Create language and execution environment from state transitions and constraints

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>L/C</td>
<td>Init</td>
<td>Init</td>
<td>Init</td>
<td>Init</td>
<td>Init</td>
<td>Issue Req</td>
<td>Issue Req</td>
<td>Issued</td>
<td>Issued</td>
<td>Issued</td>
<td>Confirmed</td>
<td>Confirmed</td>
</tr>
<tr>
<td>Cash</td>
<td>Init</td>
<td>Init</td>
<td>Init</td>
<td>Init</td>
<td>Init</td>
<td>Cash Lock</td>
<td>Cash Lock</td>
<td>Cash Lock</td>
<td>Cash Lock</td>
<td>Cash Lock</td>
<td>Settled</td>
<td>Settled</td>
</tr>
<tr>
<td>Goods</td>
<td>Init</td>
<td>Init</td>
<td>Init</td>
<td>Init</td>
<td>Init</td>
<td>Init</td>
<td>Init</td>
<td>Shipped</td>
<td>Received</td>
<td>Received</td>
<td>Received</td>
<td>Received</td>
</tr>
<tr>
<td>Contract</td>
<td>Init</td>
<td>A signed</td>
<td>B signed</td>
<td>Both</td>
<td>Both</td>
<td>Both</td>
<td>Both</td>
<td>Both</td>
<td>Both</td>
<td>Both</td>
<td>Both</td>
<td>Both</td>
</tr>
</tbody>
</table>

Not allowed to reverse
Limitation of Formal Verification

Limitation of automated tool
Upper bound of memory, …
Not sufficient for complicated protocols

How can we verify the correctness of formalization?

Formal verification does not assure the security in most cases

Need templates and languages which are suitable for formal verification
Conclusion

Applicability of Formal Analysis and Formal Verification

Current activities can help four layers of Blockchain Security

Possibility to define specific language for Application Logic Layer