Secure smart contract attestation using Intel SGX

Research Project by Agniv Chatterjee (a.chatterjee-3@student.tudelft.nl)
Supervised by Prof. Dr. Kaitai Liang
CSE3000 | Delft University of Technology
Nov '21 - Jan '22

1 Introduction

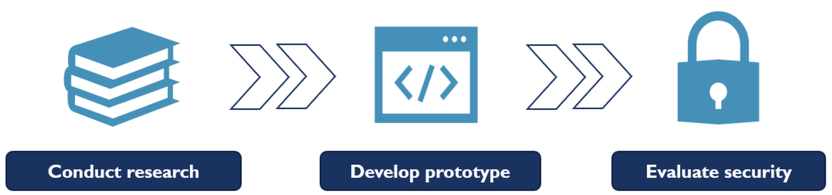
- Smart contracts are not immune to vulnerabilities and can have security issues resulting in privacy and data compromise.
- As a potential solution, TEEs can be used to run smart contract chaincode in a secure container.
- **Motivation:** Existing research makes too few attempts to develop smart contracts leveraging TEEs

2 Research Question

Q: How can Intel SGX be used to enhance security of smart contracts on Hyperledger Fabric?

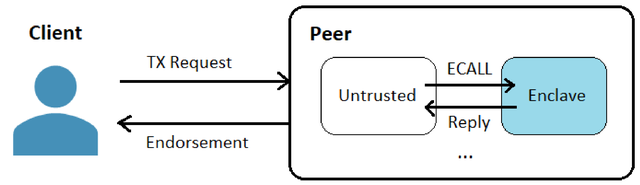
- How to apply SGX to execution of an e-voting prototype?
- What is present literature on Fabric smart contract security?
- How effective is SGX as TEE solution for smart contracts?

3 Methodology



4 Prototype

- The prototype makes use of a Fabric test network containing a client, two peers and an ordering service.
- Each peer is equipped with an enclave where the chaincode resides and produces an endorsement with the execution result.



- The smart contract allows to open/close elections, submit encrypted votes for candidates and evaluates the election winner, i.e. candidate with most votes.

```
// 1. Create an election titled 'Prime'
$ node buildElection.js organizer org1 electionPrime ben
simon jim
--> Invoke e-voting chaincode: Create a new election
--> Result: OK


// 3. Let 'voter1' submit a vote for candidate 'ben'
$ node addVote.js voter1 org1 electionPrime ben
--> Invoke e-voting chaincode: Add a new vote
--> Voter ID: CN=voter1
--> Result: Vote: 738dcfb07a4f2308677dca8c1d4ce6cb29197...
--> Result: OK


// 8. Determine the candidate with highest number of votes
$ node evaluateElection.js organizer org1 electionPrime
--> Invoke e-voting chaincode: Evaluate election
--> Result: The candidate with most votes is ben
--> Result: OK
```

```
// 4. Allow 'voter1' to query the submitted vote
$ node queryVote.js voter1 org1 electionPrime 738dcfb07...
--> Invoke e-voting chaincode: Query vote
--> Result: Vote: {"voteFrom": "CN=voter1", "voteTo": "ben"}

$ node queryElection.js organizer org1 electionPrime
--> Invoke e-voting chaincode: Get Election
--> Result: Election: {
    "name": "electionPrime",
    ...,
    "privateVotes": {
        "\u0vote\u0electionPrime\u0738dcfb07a4f...": {
            "hash": "9073790a8a849ffb6a5f0475a53532e53f6..."
        }
    },
    "publicVotes": {},
    ...
}
```

5 Discussion

- Data in enclave cannot be tampered with and remains private.
- Peers can only see the blockchain state up to what is made public.
- A malicious peer can still influence the order in which transactions are run to break confidentiality.
- Future scope of work can include comparing overhead to that of an unprotected environment.

6 Conclusion

- Intel SGX has several advantages for smart contract execution and can encrypt the result to client.
- However, there is tradeoff between security and implementation complexity for developer.
- SGX can also provide a solution for risks pertaining to transaction flow in Hyperledger Fabric.

Related literature

[1] Kazuhiro, Y., et al. 14 Mar. 2019. 'Potential Risks of Hyperledger Fabric Smart Contracts.' IEEE, https://ieeexplore.ieee.org/document/8666486
[2] Brandenburger, M., et al. 22 May 2018. 'Blockchain and Trusted Computing: Problems, Pitfalls, and a Solution for Hyperledger Fabric.' ArXiv.org, https://arxiv.org/abs/1805.08541.
[3] Brandenburger, M., Cachin, C. 1 Oct. 2018. 'Challenges for Combining Smart Contracts with Trusted Computing.' ACM Digital Library, https://dl.acm.org/doi/abs/10.1145/3268935.3268944.
[4] Zijian Bao, Qinghao Wang, Wenbo Shi, Lei Wang, Hong Lei, and Bangdao Chen. When blockchain meets SGX: An overview, challenges, and open issues. IEEE Access, 8:170404–170420, 2020.