Using Self-Encryption to safeguard data security in Fabric's smart contract

Secure Enhancement for Samart Contract

Authors

Chaiwon Park
c.park-3@student.tudelft.nl

Affiliations

Dr. Kaitai Liang
kaitai.liang@tudelft.nl

01 Introduction

With the increase of decentralised applications running on blockchains, security is becoming important aspect. Additionally, a smart contract is the most vulnerable components of the blockchain.

Self-Encryption is a unique encryption method as the key is not required in order to encrypt a file.

02 Objective

The following sub-questions will be answered to elicit the final result:

- What is Hyperledger Fabric Smart Contract?
- What is the definition of the Self-Encryption?
- How can Self-Encryption enhance the security of the Smart Contract?

03 Methodology

The following framework, library and language will be used to implement a Self-Encryption merged Smart Contract:

- Hyperledger Fabric
- MaidSafe Self-Encryption Library written in Rust
- Hyperledger Fabric Smart Contract in Javascript
- Command Line Tool for analysis

04 Contribution

Figure 1 presents the detailed workflow of the implemented prototype between the user, application, peers, orderer, smart contract, ledger and external database (IPFS).

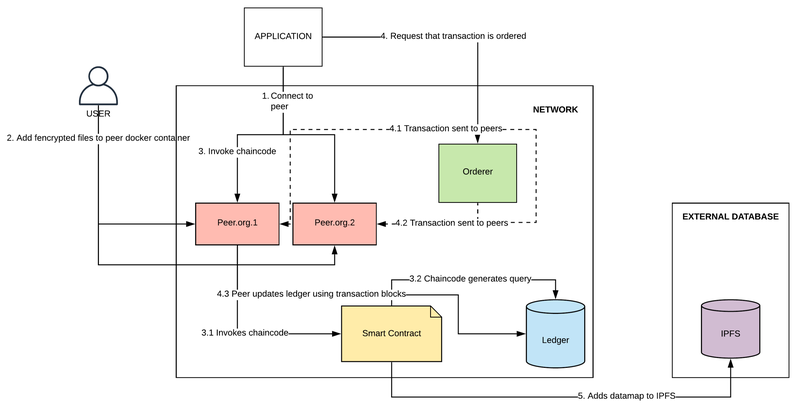
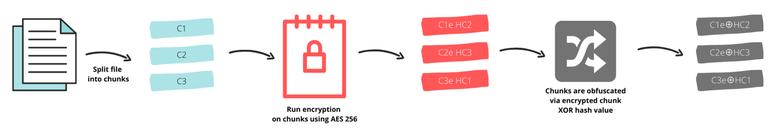


Figure 1

Figure 2 presents how self-encryption works, including what the data map is.



Data Map

Pre Encryption	Post Encryption
HC1	C1e⊕(HC2)
HC2	C2e⊕(HC3)
HC3	C3e⊕(HC1)

Figure 2

05 Results/Findings

The image below shows the results of how encrypted files and non-encrypted files are saved in the ledger.

```
# Encrypted File
"File":"
[{"filename":"3d20b45c49191b7fe0db46cbcdb3f00f12506e6ec7289de9c71955bcfa88c3d0","content":
{"type":"Buffer","data":[35,251,91,206,80,74,192,208,246,6,150,27,163,115, ... ]}}]

# CID
peer chaincode query -C mychannel -n basic -c '{"Args":["PutDataMapToIPFS"]}'
QmeHdzTp8Ebef8MJuqKTj1LmpDwCuBq77NsjsLx3PdZhi5

# Non-Encrypted File
[{ "name":"Bob", "device":"google home", "date":"2022-01-18T13:45:00.000Z",
"command":"Turn on spotify"}, ...]
```

06 Conclusion

Self-Encryption enhances the security of the data in the smart contract

However,

- Escalate the execution time and memory usage
- Add complexity to the user experience

Future work:

- Better data map management system
- Using advanced encryption

TU Delft Delft University of Technology