Michael Sneberger – ASU ID: 1000001544
Individual Project Report
Derived from
Security Assurance for Smart Contracts
Group 8 Project Report - CSE 543 - Fall 2019

Overview of the Group 8 Project Report:

Group 8 undertook an investigation of what is a blockchain, what kinds of blockchain platforms are available and which platforms support smart contracts, what is a smart contract, and how parties to a smart contract on top of a blockchain platform can be assured of their veracity and security.

As Group Leader, and someone who likes to start early on projects, I undertook to create a skeleton outline of the Group 8 Report. Fortunately, the first reference I read was a 2017 article by Maher Alharby and Aad van Moorsel which was survey of scholarly blockchain/smart contract articles that listed issues that should be included in any future review of the subject. [1] This was exactly on point with Group 8's task, and with some adjustment I created a Google Document that presented an outline of eight sub-headings to be addressed based on this article, which allowed each member of the group to choose a particular section, with the goal of each member writing about their own section.

As Group Leader I ultimately wrote all the introductory materials of the report, and based on the individual section conclusions, the conclusion. After pushing the Group members to complete drafting their sections, I spent many hours proofreading the Report, making corrections, and suggesting edits to group members that would improve the Report. Finally, along with help from Sarthak Shetty, the List of References, and citations to those references were placed in the order in which they appear in the Report in order to finalize efforts to assure that the Report reads as one cohesive document.

Lessons Learned from the Group 8 Report:

Group 8 and myself learned a lesson by reviewing the 28 questions that were presented for us to answer during our class presentation. What we learned is that even though we understood that we were narrowing the focus of out inquiry to the Ethereum blockchain platform, we could have made that clearer in order to provide guidance to our questioning classmates, and to allow us to better defend the Report. We should have added this sentence to the above quoted Goals and Scope section: "Given the breadth of the topic of blockchain and smart contracts, we have narrowed the scope of this Report down to smart contracts on the Ethereum blockchain platform."

I also learned that it is difficult as a leader to motivate team members when you do not have the power to affect their career or income. I decided early that my best path would to lead by example and as a result attempted to work harder than any other team member, and do so early in order to provide motivation.

Introduction to the Group 8 Report:

Now that more and more commerce is being undertaken online, between people and organizations in disparate locations, over publicly accessible channels, the antiquated process of affixing "wet" signatures in ink to paper contracts can be accomplished more efficiently with smart contracts, and smart contracts are often facilitated and memorialized by an underlying blockchain technology platform. Smart contracts often require the ability to write specific contract terms by way of a complete language sitting atop a blockchain protocol.

A blockchain is an open distributed ledger that can record transactions between two parties efficiently and in a verifiable and permanent way. Smart contracts inherit properties of underlying blockchains which include an immutable record of data, and the ability to mitigate single points of failure. Smart contracts can also interact with each other via calls. Unlike traditional paper contracts that rely on middlemen and third-party intermediaries for execution, smart contracts automate contractual procedures, minimize interactions between parties, and reduce administration costs

A contract – smart or otherwise – needs to be adopted formally by the parties that they are bound by the promises made in the contract. A digital signature is a digital representation that mimics the process of manually signing a contract with a signature in ink – but meet even more stringent criteria for an effective signature.

The Turing-Complete Solidity language on the Ethereum blockchain platform allows developers to program their own smart contracts. Smart contracts have been used for many applications.

Due to the inherent structure of the blockchain itself or due to the coding practices followed by the developers writing the code that facilitates smart contracts, developer can face multiple problems and issues while developing smart contracts for the Ethereum blockchain platform.

Given that smart contracts can and do represent significant value to the parties, vulnerabilities in smart contracts present serious issues, and require fixing at early onset for many reasons, not the least of which that bugs in smart contracts, once they are developed, cannot be fixed because smart contracts are immutable in nature. Thankfully, there are multiple avenues for assuring the integrity of smart contracts.

Ultimately, smart contracts utilizing the structure of a blockchain platform are already in use, and will surely provide the efficient future of contracting and tracking in many areas of human endeavor, and the World will rely on computer scientists to design and facilitate these contracts in a secure manner.

DESCRIPTION of BLOCKCHAIN:

A blockchain is an open distributed ledger that can record transactions between two parties efficiently and in a verifiable and permanent way. Open means "accessible to all" and public ledger means history of transactions which is available to everyone in the network i.e., all the nodes have local copies of all the transactions. Permanent way represents that the transactions registered once cannot be erased and anyone cannot tamper them.

Structure of a Blockchain:

The base of blockchain is Merkle tree structure and hash associated with it. Merkle tree basically stores all the hash values of the appropriate child nodes. These hash values are propagated throughout all the nodes till leaf nodes, with the leaf nodes representing blocks in the blockchain.

Each block has two parts called block body and block head which has four significant parts:

- Nonce
- Time stamp
- Hash of previous block
- Merkle Root hash

And block body contains the records of the transactions stored in that block. If any transaction is tampered then there will be hash difference will be identified very easily.

Working of a Blockchain:

How transactions are grouped together to form a block and how blocks are chained to form a blockchain. To do this work, there are some set of consensus algorithms which nodes in the network should follow to formulate their work. We can broadly classify these algorithms into:

- Proof of Work (PoW)
- Proof of Stake (PoS)
- Proof of Elapsed Time (PoET)

ETHEREUM and SOLIDITY are most widely used platform for SMART CONTRACTS

Ethereum was developed in late 2013 by Vitalk Buterin. It is open sourced and blockchain based distributed computing platform. Ethereum runs smart contract on a custom built blockchain. Ethereum Virtual Machine (EVM) offers run-time environment to smart contracts which are programs that manage the performance of accounts that are inside the Ethereum state. Each node inside the network need to execute an EVM implementation. Solidity is a full-function, object oriented programming language which is used to code smart contracts on the Ethereum platform.

Ethereum was been built for restrained access versus mass consumption. It is also a Proof of Work (PoW) based platform that is relatively slower in terms of speed. The native cryptocurrency of Ethereum is Ether. It is used for fueling the Ethereum ecosystem [2]. To develop applications, the developers need to pay in terms of Ether in order to execute transactions and run apps on the Ethereum network.

DESCRIPTION OF A SMART CONTRACT

The history of smart contracts can be traced back to the 1990s when Wei Dai, a computer engineer created a post on anonymous credits, which described an anonymous loan scheme with redeemable bonds and lump-sum taxes to be collected at maturity. [3] Today, with the

development of blockchain technology, smart contracts are being constructed as programs running on blockchain nodes and can be issued among untrusted, anonymous parties without the involvement of any third party. [3] A smart contract is a computer protocol intended to digitally facilitate, verify, or enforce the negotiated performance required under a contract. Smart contracts allow the performance of credible transactions without third parties.

DESCRIPTION of DIGITAL SIGNATURES

As its name implies, a digital signature is a digital representation that mimics the process of manually signing a contract with a signature in ink. [4] Given that "wet" signatures in ink can be forged, or the contract to which a "wet" signature is affixed can be altered without the signer knowing, an unalterable digital signature provides a more secure way to show that a party has indicated their agreement to adhere to the terms of a particular contract. [4] More specifically, while a "wet" signature on a paper document cannot meet the five criteria for an effective signature, a digital signature scheme can meet those criteria: Authenticity; Unforgeability; Non-reusability; Inalterability of the underlying document; and The inability for the signature to be repudiated by the signer. [5]

A modern digital signature scheme uses asymmetric cryptography with a public and private key to prove that a party signed a unique contract. [4] In its most general sense, this is done by the signor encrypting a document with her private key, then anyone being able to decrypt the resulting ciphertext with a public key, thus proving the signer signed the specific document.

Asymmetric cryptography is the basis of a digital signature.

In 1976, inspired by the work of Merkle, Diffie and Hellman derived the asymmetric concept of a public and private keys as a solution for securely exchanging a symmetric encryption/decryption key over a public, unsecured channel. [6] The Diffie-Hellman scheme involves the traditionally named Alice and Bob wanted to use symmetric cryptography in order to keep messages passed between then secret. In order to do so, they need to exchange a symmetric key that each of them will use to both encrypt and decrypt messages to and from each other. Securely exchanging this symmetric key is the aim of the Diffie-Hellman scheme.

In 1978, Rivest, Shamir and Adelman (RSA) built upon the work of Diffie-Hellman to develop the basis of a modern digital signature by adding authentication by way of signature using private keys to thwart a man-in-the-middle attack. This bolsters authentication by way of having the signer of a document first hash the document, then sign/encrypt that document hash with their private key. This would allow any third party to use the signer's public key to decrypt the message, then check the hash of the signed documents against the document itself thus verifying both that the signer is the only person who could have – and did – sign the contract. [7]

How do we know who signed an Ethereum blockchain-based smart contract?

In addition to Bitcoin, the Ethereum blockchain platform uses a digital signature scheme that builds on the overall theme of RSA, but uses elliptic curve cryptography (ECC) to derive its keys as opposed to RSA's reliance on the multiplication of two prime numbers. Elliptic Curve Digital Signature Algorithm (ECDSA) provides similar service as other digital signing algorithms, but using ECC provides ECDSA with greater efficiency, particularly requiring smaller keys to provide equivalent security.

Conveniently, public and private ECC/ ECDSA keys are generated when a user creates a new Ethereum account. These initial keys can then be used to support smart contracts because in addition to providing access to the Ethereum ledger, the private key serves as a basis for digital signatures (while a hash of the public key serves as the Ethereum address of the ledger, and ECDSA digital signatures in Ethereum also support smart contracts outside of the blockchain.

FORMAL VERIFICATION PROTOCOLS

In order to build robust smart contracts and verify their safety against external attacks, it is essential to take into account the blockchain's behavior properties. A novel modeling formalism with strong semantics for smart contracts and blockchain properties can been used.

Applying formalism on a concrete example of a smart contracts, modeling its behavior and

interactions with the Behavior Interaction Priorities framework and the analysis of its results using Statistical Model Checking Tool can reveal scenarios where the smart contract behavior can be breached by hackers.

APPLICATIONS of SMART CONTRACTS in the REAL WORLD

Digital Identity; Banking; Tax Records; Insurance; Real Estate and Land Titles Recording; Supply Chain; IoT (Internet of Things); Authorship and Intellectual Property Rights; Life Science and Health Care; Gaming and Gambling; Finance; Government; Health-care; and Food Safety:

ISSUES FACING DEVELOPERS WRITING SMART CONTRACTS

The Majority Attack; Forking inconsist-encies (due to updates in consensus rules); The scale of the blockchain; Current regulations and rules; Cost of infrastructure; Reentrancy on a single function; Front running; Integer Overflow and Underflow; Using revert functions; Insufficient gas griefing (attackers can pass another void contract into a contract in the chain. When the main contract calls the void contract and it fails, the main contract could also fail).

MAKING SMART CONTRACTS SECURE:

FSolidM framework:

Most past attacks on smart contracts are mostly because of semantic gaps in assumption of smart contracts. For example, reentrancy attack [8] where attackers invoke functions repeatedly before first function call is completed, in order to take control of the flow of the chain. FSolidM [9] is a graphical tool for rigorous designing of finite state architecture in Smart contracts.

A specification language for smart contracts (SPESC):

While designing and implementation restricted to developers and designers limit the intelligence in designing smart contracts which later cause vulnerability issues. SPECS is a tool that challenges this restriction and unveils new levels of designing involving non-IT smart people from the industry in designing perfect smart contracts. SPESC framework uses natural language in writing smart contracts.

CONCLUSIONS/RECOMMENDATIONS:

Smart contracts on top of a blockchain platform can be formally verified. Modeling formulas have been applied, and have shown the general reliability of smart contracts. And that formal modeling can highlight issues facing developers writing smart contracts.

There are ways to assure the security of smart contracts. Certain tools, like the FsolidM framework, can assist smart contract designers/ developers to design semantically secure smart contracts as finite state machines with an aim to keep in mind every possible action user using this smart contract can take and to handle them.

REFERENCES

[1] Maher Alharby, Aad van Moorsel. 2017. Blockchain-based Smart Contracts: A Systematic Mapping Study. In: Fourth Int'l Conf. on Computer Science and Information Technology (CSIT-2017), 2017.

[2] Akash Takyar. 2019. Top Blockchain Platforms of 2019 for Blockchain Application - LeewayHertz. (October 2019). Retrieved from: https://www.leewayhertz.com/blockchain-platforms-for-top-blockchain-companies/

[3] Yining Hu, Madhusanka Liyanage, Ahsan Mansoor, Kanchana Thilakarathna, Guillaume Jourjon, Aruna Seneviratne. June 8, 2019. Blockchain-based Smart Contracts - Applications and Challenges

[4] Jonathan Katz. 2010. Digital Signatures. Springer, New York, NY.

[5] Bruce Schneier. 1996. Applied Cryptography. (2nd. ed.) Wiley, Indianapolis, IN.

[6] Whitfield Diffie, and Martin Hellman. (1976). New directions in cryptography. IEEE Trans. Inform. Theory IT-22, 6 (Nov. 1976), 644-654.

[7] R. L. Rivest, A. Shamir, and L. Adleman. 1978. A method for obtaining digital signatures and public-key cryptosystems. Commun. ACM 21, 2 (February 1978), 120-126.

[8] Anonymous. Know attacks in smart contracts. Retrieved from: https://consensys.github.io/smart-contract-best-practices/known_attacks/

[9] Anastasia Mavridou, Aron Laszka. 2018. Designing Secure Ethereum Smart Contracts: A Finite State Machine Based Approach. In: Meiklejohn S., Sako K. (eds) Financial Cryptography and Data Security. FC 2018. Lecture Notes in Computer Science, vol 10957. Springer, Berlin, Heidelberg