Brown University

Yana Hrytsenko

Final Project: “Blockchain for PKI: Using Blockchain data structure for Public Key Infrastructure”

May 2017

Blockchain for PKI

Using Blockchain data structure for Public Key Infrastructure (PKI)

Introduction:

Nowadays, the term “encryption” is frequently used but the process is not. Many consider encryption to be somewhat complex and an unnecessary extra step and therefore decide to skip it. However, we should all understand how important it is to encrypt the message contents when communicating with another user. Justification comes from the results of the latest election when John Podesta, who was in Hillary Clinton's team, had his email hacked. A lot of sensitive information was leaked to the public, and this heavily influenced the outcome of the elections.

Public key cryptography is used to encrypt a message and safely deliver it between two users. In order for a user to encrypt a message, he has to know the public key of the recipient because it has to be used in the encryption process.

In order to make encryption less inconvenient for a user, I propose to use a blockchain data structure, such as what Ethereum cryptocurrency system uses. In order to store a record of the user’s email and his public key, we can use smart contracts. Any user, who would like to find out the public key of recipient, would be able to query a blockchain and receive the information safely and efficiently.

I will also consider the possible obstacles for querying a blockchain and receiving correct information, and will suggest a possible solution.

1. Using blockchain data structure for PKI

Blockchain is a type of data structure, where each building block contains a hash pointer to a previous block. The hash pointer stored in the header of each block is a "summary" of the previous block, which ensures that the data written into the blocks cannot be tampered with [1], [2]. This makes blockchain a secure database for storing public key information. There is a lot of past and currently active research dedicated to the security of blockchain, which show that blockchain is cryptographically proven to be secure.

2. Bitcoin and Ethereum

Blockchain is widely used in Bitcoin, Ethereum and many other cryptocurrencies, where this data structure is the central mechanism of secure financial systems. Ethereum, is a cryptocurrency, which is different from other cryptocurrencies in a way that it uses Turing-complete script, which allows for more possibilities than just mining currencies [3], [4]. Just as Bitcoin, Ethereum uses blockchain for storing and processing the information on the network. I will be looking at Ethereum in particular because of their use of smart contracts, which will allow us to write a program that will be used to create a PKI.

3. Smart Contracts and Solidity Script

One of the special features that separates Ethereum from the rest of altcoins, is the use of smart contracts. A smart contract is a small program that a user can write and upload on the blockchain for a fee. This program, after being executed by a Ethereum-specific virtual machine, will perform a set of tasks defined by the user [5].

In order to write a contract on Ethereum, users may use Solidity, which is a Turing-complete high level programming language used to create smart contracts.

4. Defining a contract for storing a public key of the user

In order to create a public key infrastructure on the blockchain, we have to define the contracts in a way such that several rules are followed:

- The record stored has to be of public key/email format
- emails and keys have to be unique
- The user must pay a fee for creating a contract

Smart contracts are very flexible in regard to what kind of procedure it will perform. Since Solidity is a Turing-complete language, the user can define a program that will achieve many different tasks [6]. Even though we could create many features in the contract, we would only need two or three for our purposes. Our goal is to create a record of a unique public key and a unique email address record in a database. The idea is that the uniqueness of the record will allow the mapping of the email to the public key in the same way that the data is mapped to a primary key field in a hash map. Instead of associating a PK with a particular email address, we can do the opposite; we can associate any email address with a given PK. This will be easy to do because each user in the network already has a PK associated with his identity. Therefore, the user can register an email address associated with the public key used in the network. This way the user himself is responsible for providing the correct email address since the public key of the user is already known and can be easily verified.

We can look at the following scenario: a user creates a “PKI business”, a database of PK/email addresses. We know that the user on the Bitcoin/Ethereum has some amount of coins in his possession already. The owner may provide a service of “identity verification” where he physically checks the identity of the PK/email owner to assure the person who requested the information, that this particular PK/email address indeed belongs to a particular person. In order to make this business reputable and safe for the clients the owner of this business will use his own funds to guarantee the credibility of the information. In other words, he will be responsible for the accuracy of the information on this website/database. If anyone can prove him wrong he will be obligated to give up the total amount of coins owned by him to the user who proves the information to be false. This way the resources of the owner are at stake, and therefore he is motivated to provide honest information to the user who sends the query. This would work particularly well with the current Bitcoin exchange rate, where each Bitcoin is worth $1200. Having to give up even a few coins would be very costly, and having to give up all of one’s coins would be especially hurtful.

Another important feature of smart contracts is the fee that the user must pay in order to upload his program on the blockchain. We can use this feature in order to collect, withdraw and transfer the funds as needed for this PKI to operate efficiently. This will be discussed in detail in section 7.

As an example, we can write the following program that stores a pk/email record in a contract for a fee of at least 10 wei (wei is the currency in Ethereum):

```
contract PublicKeyRegistry{
     mapping(bytes32 => key) public pkTable;
     function assignEmail(bytes32 email){
          if(msg.value < 10){
               throw;
          }
          if(pkTable[email] == 0){
               pkTable[email] == msg.sender;
          }
     }
}
```
[7],[8].

The above program creates a public key registry. The email of the user is mapped to the public key. The `assignEmail` function accepts the email of the user and checks that the user payed the fee for execution of the contract. Any user can call a function on Ethereum contract, but each call is signed by the user id. Therefore, it will be easy to check who the caller is, which turns out to be a very convenient feature for our purposes.

Another useful procedure we can write for our PKI is the following program which allows the money paid by the owner of the program, to be send to another user id:

```
    function transfer(address _to, uint256 _value) {
        if (balanceOf[msg.sender] < _value) throw;
        if (balanceOf[_to] + _value < balanceOf[_to]) throw;
        balanceOf[msg.sender] -= _value;
        balanceOf[_to] += _value;
        Transfer(msg.sender, _to, _value);
     }
```

The above program does the following:

- Checks if the sender has enough funds allocated in the contract
- Checks for overflows
- Subtracts funds from the sender
- Adds the same amount to the recipient
- Notifies anyone listening on the network that this transfer took place [9].

The above function can be incorporated in the contract defined earlier, which will expand its functionality. In general, the user is able to add as many features as he would like to the contract he defines [10].

5. Difference between “full nodes” and “lightweight nodes”

Nodes are the backbone of any network system, especially in Ethereum where the communication relay and consensus is reached through the communication between these nodes. There are two kinds of nodes: full nodes and lightweight nodes [11]. Full nodes do all the “heavy work”. Their responsibilities include the following:

- accepting transactions from other full nodes
- accepting blocks from other full nodes
- validating transactions
- validating blocks
- communicating with other full nodes in the network and transferring all the information after validation step further to other full nodes

To become a full node, a user needs to:

- store a complete copy of the blockchain
- synchronize the blockchain information often with other full nodes in order to keep the information updated
- purchase hardware that runs the latest version of Linux, Windows or OS X
- have 125 gigabytes of free disk space available
- have 2 gigabytes of memory (RAM) available
- have broadband Internet connection with upload speeds of at least 400 kilobits (50 kilobytes) per second
- Have access to an unmetered connection, a connection with high upload limits, or a connection you regularly monitor to ensure it doesn’t exceed its upload limits.
- Have at least 6 hours a day that your full node can be left running (preferably as many as possible) [12].

Lightweight nodes, as the name suggests, do not need to go through so many steps in order to take advantage of using blockchain technology and communicate with the network.

Lightweight nodes do the following:

- download only a small portion of the blockchain that relates to a particular user's transactions
- connect with full nodes from which they will be receiving all the basic information about the blockchain

- access the index header ordering system for each block that is used by the blockchain (this gives each lightweight node high level details about their own transactions) [13], [14].

Why do we need more full nodes:

As we can see, the lightweight nodes do not need to invest so much into working with blockchain and communicating with the network. Therefore, the number of lightweight nodes grows much faster and the number of full nodes declines instead. This disproportion puts a lot of heavy work on the full nodes. A very low number of full nodes on the network puts the safety of the system at risk. The less full nodes there are, the more Sybil attacks are likely to happen, where fake nodes are created to collect the incentives or data from the network [15]. With too many fake nodes the network may slow down. The reason is that the legitimate full nodes and lightweight nodes will still try to communicate with these fake nodes.

To be a full node there are no incentives if the node is not a miner but rather an altruist who works for the idea of helping to maintain a healthy network environment.

For our purposes, we would like a slightly different model of communication between lightweight nodes and full nodes. In particular, we would like the lightweight nodes to be able query full nodes for a user’s public key, through whatever API the system provides. Since lightweight nodes do not store the copy of the full blockchain and cannot simply look up the key themselves, the full nodes will be the ones to do this job.

For a full node to perform such task, it would be very costly: uploading a blockchain once, synchronizing it periodically with other full nodes on the network and executing some other tasks entail large costs of electricity and hardware, as well as high bandwidth of the internet [16]. As mentioned earlier, if the full node is not mining, besides all the checking and communicating the information to other nodes, then he does not get paid and all the costs come from his own pocket [17],[18]. While it is known that many are willing to work towards the common goal on the basis of good will, it would surely be easier to make full nodes answer queries if they were financially incentivized.

7. Incentivizing full nodes to answer the queries

The idea of financially incentivizing full nodes to do their hard work is not new. There have been several proposals and programs that would pay some amount of cryptocurrency to the registered full nodes who were active on the network. The following programs were examples of such incentives.

- A trial incentive program, funded by the Bitcoin foundation Bitnodes, required full node owners register details about their nodes in order to receive incentives. Incentives were paid weekly to the nodes chosen at random.
- Microsoft Research introduced a proposal on the incentives program for information propagation in a way that the loss of incentives to fake nodes would be avoided. The proposal was to draw the rewards from the mining rewards and the transaction fees.

- Another group proposed incentivizing the creation and the operation of full nodes by requiring the lightweight nodes, who receive information from full nodes, to pay a “service fee”.
- DASH (formerly Darkcoin) used a two-tier network, where the “masternodes” would get rewarded for running their servers for network operation.
- Spreadcoin (SPR) used a similar approach to DASH with two-tier network, where the “ServiceNodes” were incentivized for their operation [19], [20], [21].

The main goal of the above programs was to incentivize creation of full nodes that would in turn help to maintain healthier and more secure network.

Even though it still would surely be helpful to create healthier network, our goal is to incentivize the full nodes who are the backbone of the Ethereum network so that they can answer the queries that were submitted by the lightweight clients. Checking the integrity of the data on the blockchain and mining cryptocurrency are usually the main responsibilities of full nodes. In addition to these tasks, we want the full nodes to look up the information in the blockchain, retrieve the public key of the user and output the results of the query. The question we have to ask is: why would they want to do this extra work if it does not give them any financial reward or help the system in general? As far as the full nodes are concerned, it is just an extra step they are asked to perform for someone's benefit. Now we can look at a possible solution to this problem: incentivizing full nodes to answer the queries.

As mentioned earlier, when creating a contract, the user must pay a fee in order for the contract to be executed and published on the blockchain. Currently, as it is written in the rules of Ethereum, the fee is at least 10 wei and the money just stays there forever. We could use this money to pay the full nodes for their work [22].

We could do the following: once the contract is written and uploaded on the blockchain, the program checks if at least the minimum of 10 wei fee has been paid by the user who created the record of his public key. If such fee has been paid, then the contract will be executed. Next, this money will go to a special output address, which will be just a financial pool where all the fees will be collected and distributed in the future to registered full nodes who have been active for a specific period of time.

The incentives would be distributed at random to the set of nodes who are registered as full nodes. For example, if there are ten nodes registered as full nodes [23], the algorithm would pick one node at random and transfer a specified amount of coins to his address.

When the time comes to reward another full node for his work, there will be only nine nodes to chose from since one of the ten nodes have been rewarded already. This will ensure that each node will get rewarded eventually, and no node will be rewarded more than once until all of the nodes have received the incentives. There is no particular schedule for the reward because the nodes are chosen at random, which makes it a fair reward schedule. This would also solve a potential problem of a node artificially sending answering queries, and then be rewarded every time due to having the highest number of queries answered.

References

[1] A. Narayanan, J. Bonneau, E. Felten, A. Miller, S. Goldfeder, “Bitcoin and cryptocurrency technologies: a comprehensive Introduction”, Princeton and Oxford: Princeton University Press, 2016, p.65.

[2] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," October 2008 [Online] Available: https://bitcoin.org/bitcoin.pdf [Accessed: February 2nd].

[3] A. Narayanan, J. Bonneau, E. Felten, A. Miller, S. Goldfeder, “Bitcoin and cryptocurrency technologies: a comprehensive Introduction”, Princeton and Oxford: Princeton University Press, 2016, p.263.

[4] Eva, “A 101 Noob Intro to Programming Smart Contracts on Ethereum,” October 2015 [Online] Available: https://consensys.github.io/developers/articles/101-noob-intro/ [Accessed: March 22nd].

[5] A. Narayanan, J. Bonneau, E. Felten, A. Miller, S. Goldfeder, “Bitcoin and cryptocurrency technologies: a comprehensive Introduction”, Princeton and Oxford: Princeton University Press, 2016, p.264-265.

[6] “Contracts and Transactions: account types and transactions,” August 2015 [Online] Available: https://github.com/ethereum/go-ethereum/wiki/Contracts-and-Transactions [Accessed: February 13th].

[7] A. Narayanan, J. Bonneau, E. Felten, A. Miller, S. Goldfeder, “Bitcoin and cryptocurrency technologies: a comprehensive Introduction”, Princeton and Oxford: Princeton University Press, 2016, p.265.

[8] Kalodner, Harry, M. Carlsten, P. Ellenbogen, J. Bonneau, and A. Narayanan. "An Empirical Study of Namecoin and Lessons for Decentralized Namespace Design," Princeton University, [Online] Available: http://randomwalker.info/publications/namespaces.pdf [Accessed: April 12th].

[9] A. Zsales, D. Ames, Georgem, Chaositech, WoodenBush, Coins101, “Proof of Bitcoin Node: a mechanism for a Bitcoin Full NOde Incentives & Bitcoin Mining Rewards Program”, October 2015, p.29.

[10] “Comparison of Namecoin to other projects,” [Online] Available: https://namecoin.org/docs/faq/ [Accessed: March 14th].

[11] A. Zsales, D. Ames, Georgem, Chaositech, WoodenBush, Coins101, “Proof of Bitcoin Node: a mechanism for a Bitcoin Full NOde Incentives & Bitcoin Mining

Rewards Program”, October 2015, p.15.

[12] “Running A Full Node: Support the Bitcoin network by running your own full node,” 2017. [Online] Available: https://bitcoin.org/en/full-node#special-cases [Accessed: March 16].

[13] A. Zsales, D. Ames, Georgem, Chaositech, WoodenBush, Coins101, “Proof of Bitcoin Node: a mechanism for a Bitcoin Full NOde Incentives & Bitcoin Mining Rewards Program”, October 2015, p. 12-13.

[14] “Light Client Protocol,” [Online] Available: https://github.com/ethereum/wiki/wiki/Light-client-protocol [Accessed: May 2nd].

[15] “Running a full node - Bitcoin,” 2017. [Online] Available: https://bitcoin.org/en/full-node#ubuntu-1610 [Accessed: April 10th].

[16] A. Zsales, D. Ames, Georgem, Chaositech, WoodenBush, Coins101, “Proof of Bitcoin Node: a mechanism for a Bitcoin Full NOde Incentives & Bitcoin Mining Rewards Program”, October 2015, p.21.

[17] A. Narayanan, J. Bonneau, E. Felten, A. Miller, S. Goldfeder, “Bitcoin and cryptocurrency technologies: a comprehensive Introduction”, Princeton and Oxford: Princeton University Press, 2016, p.69-72.

[18] N. Houy, "The economics of Bitcoin Transaction Fees," February 2014 [Online] Available: https://hal.archives-ouvertes.fr/file/index/docid/951358/filename/1407.pdf [Accessed: March 7th].

[19] A. Zsales, D. Ames, Georgem, Chaositech, WoodenBush, Coins101, “Proof of Bitcoin Node: a mechanism for a Bitcoin Full NOde Incentives & Bitcoin Mining Rewards Program”, October 2015, p.34.

[20] J. Redman, “Should Full Bitcoin Nodes Get Rewarded like Miners?,” February 2016 [Online] Available: https://news.bitcoin.com/full-bitcoin-nodes-get-rewarded-like-miners/ [Accessed: February 20th].

[21] J. Dalais, “Bitnodes Project Issues First Incentives For Node Operators,” March 2015 [Online] Available: https://bitcoinmagazine.com/articles/bitnodes-project-issues-first-incentives-node-operators-1426544155/ [Accessed: April 17th].

[22] A. Narayanan, J. Bonneau, E. Felten, A. Miller, S. Goldfeder, “Bitcoin and

cryptocurrency technologies: a comprehensive Introduction”, Princeton and Oxford: Princeton University Press, 2016, p.265.

[23] L. Coleman, “Developers Propose ‘Proof of Bitcoin Node’ To Reinvent Bitcoin Mining Into Big Data Mining,” October 2015 [Online] Available: https://www.cryptocoinsnews.com/developers-propose-proof-bitcoin-node-reinvent-bitcoin-mining-big-data-mining/ [Accessed: April 3rd].