DAT 510: Assignment 3

Submission Deadline: 23:59, Friday, Nov. 10, 2023

Block Chain and Cryptographic Security

Assignment Objectives

The objective of this assignment is to understand and implement the fundamental principles of cryptography within the context of blockchain technology. Students will explore the role of cryptography in securing data and transactions in blockchain systems, gaining practical knowledge and skills in encryption and digital signatures.

Design and Implementation

Develop a functional blockchain system from scratch using the Python programming language. Design the data structures, create transactions, create blocks, implement a consensus mechanism (e.g., proof of authority), and incorporate cryptographic techniques for security.

Instead of developing a full-fledged peer-to-peer network to test blockchain technology, this assignment adopts a simplified yet practical approach by introducing a 'Node' class with attributes and functions to define its properties and behaviour, respectively. Multiple instances of this 'Node' class will be created to represent individual nodes within the blockchain network. Bonus: GUI for each node and using Python threading or multiprocessing modules for parallel execution of nodes will be a perfect simulation for this system.

Due to Proof of Authority as consensus, some nodes need to be marked as authority. This approach allows us to focus on the core principles and mechanics of blockchain technology and cryptography while efficiently simulating the interactions and behaviours of decentralized network participants.

Data Structure for transaction

A typical transaction in a blockchain includes the following components:

1. Sender's address or public key.
2. Recipient's address or public key.
3. Transaction data or amount or value.
4. Transaction time stamp.
5. Digital signature (use RSA for security and authenticity from Assignment 2).
6. Other info related to verification and validation.

Transaction Management

Implement transaction management within the blockchain, allowing for the creation, verification, and validation, which is the inclusion of transactions in the blockchain. transactions are often signed and verified using asymmetric cryptography RSA, which involves a pair of keys: a public key and a private key. Here's a more detailed explanation of how public and private keys are used for signing and verifying transactions.

1. Transaction Signing: When a user initiates a transaction on the blockchain, they can use their private key to sign the transaction data. This signature is specific to the transaction data and the user's private key. It proves that the owner of the private key has authorized the transaction and ensures the transaction's authenticity.
2. Transaction Verification: When nodes on the blockchain network receive a transaction, they also receive the digital signature associated with that transaction. Nodes on the blockchain network can verify the authenticity of transactions by using the sender's public key to decrypt the transaction's signature. If the decrypted signature matches the transaction data, the transaction is considered verified. Each transaction needs to be verified from all nodes in the network.
3. Transaction Validation: If the signature is verified from all nodes and the transaction data is intact, the transaction is considered valid and can be added to the blockchain by authoritative nodes. If the signature doesn't match or if the data has been tampered with, the transaction is rejected as invalid. A block should be created when there are a certain amount of pending but verified transactions.

Data Structure for Block

Start by defining the basic data structure of a block. A block typically includes the following components:

1. Index or Height: A unique identifier for the block within the blockchain.
2. Creator public key (for verification from other nodes).
3. Previous Hash: The cryptographic hash (SHA256) of the previous block, which links blocks together.
4. Timestamp: The time when the block is created.
5. Data: The actual data you want to store in the block.
6. Hash: The cryptographic hash (SHA256) of the block's content, including the previous block's hash.
7. Signature (using private key) of the authority who created the block.

8. Mine is a resource-intensive and computationally challenging problem skip this step.

Block verification

As new blocks are added to the blockchain, you can use RSA-based digital signatures to ensure that the block's content has not been tampered with during transmission. The block creator can sign the block with their private key, and other nodes can verify the signature using the public key of the creator.

Implementing the Consensus Mechanism (Proof of Authority)

Proof of Authority (PoA) is more centralized and relies on a limited number of trusted authorities to maintain the network's integrity. PoA is often used in private or consortium blockchains where participants are known and trusted.

1. In a PoA network, a predetermined set of validators or authorities are responsible for validating and creating new blocks. These authorities are typically entities or individuals with a high level of trust within the network. They are known and accountable for their actions.
2. When a transaction is submitted to the network and verified by other nodes, the authorities review and validate it. They ensure that the transaction adheres to the network's rules and is legitimate. If the transaction is valid, the authorities include it in a new block.

Assessment

GUI for each node will be a perfect illustration. But output using a simple command prompt is also acceptable. Design/coordinate interactive output for the assessment of the following features.

Create a menu which allows for the following features: Select one of the following options in Menu 1:

1. Node selection.
2. Print all verified transactions.
3. Print all pending transactions (yet to be verified. Should show who verified it, and who has not verified it).
4. Print all verified blocks in the blockchain.
5. Print all pending blocks (newly created blocks, not yet verified by all nodes in the network)

6. Exit Program

Select one of the following nodes to perform actions in Menu 1.1:

1. Node 1
2. Node 2
3. .....
4. Exit to menu 1.

Display node information. Select one of the following actions for the selected node in Menu 1.1.1:

1. Create a transaction.
2. Verify a transaction.
3. Create a block if the transactions are valid (if the node is authoritative)
4. Verify block in the blockchain.
5. Print transactions created by this node.
6. Print pending transactions that this node has not yet verified.
7. Exit to menu 1.1.

Create your own interactive prompt for the above actions to be done. Take input from the user for transaction data. In verification, show the user that transaction data and data from the signature is identical, then verify the transaction. While creating the block, show to the user that all pending transactions are verified from all nodes, then create the block. In block verification, like transactions, show both data to the user and then verify the block.

Assignment Approval (by TA and SA)

Assignment approval will have a weight on your grade for the assignment. If you are not going to get the approval before the deadline, your assignment will not be evaluated and you will fail the assignment.

What needs to be done to get the approval of the assignment:

1. Show all parts of assignment are working i.e, show the code with proper comments, results.
2. The code should have a proper README file that describes the contents of the directory and any special instructions needed to run your programs (i.e. if it requires and packages, commands to install the package. describe any command line arguments with the required parameters).
3. Source code submitted for the assignment should be your own code. If you have used sources from the internet everything should be added to the references. If you used someone's code without reference, that will also be treated as plagiarism.
4. Provide the references in the code and Report, show these parts for TA's and Student Assistants.
5. You **CAN** use available libraries/packages/classes for implementing the core functionality of the assignment.

Use **Python** as a programming language for the implementation of this assignment.

Assignment Submission

Deadline: 23:59, Friday, Nov. 10, 2023 (submit your assignment through canvas)

Final submission:

1. Source Code
   - Source code submitted for the assignment should be your own code. If you have used sources from the internet everything should be added to the references. If you used someone's code without reference, that will also be treated as plagarism.
   - Source code should be single, compressed directory in .tar.gz or .zip format.
   - Directory should contain a file called README that describes the contents of the directory and any special instructions needed to run your programs (i.e. if it requires and packages, commands to install the package. describe any command-line arguments with the required parameters).
   - You should **NOT** use available libraries/packages/classes for implementing the core functionality of the assignment.
2. A **separate** report with PDF format
   - Texts in the report should be readable by human, and recognizable by machine;
   - Other formats will **NOT** be opened, read, and will be considered missing;
   - Report should follow the formal report style guide in next page.
   - Each student should write an individual report. Each report will be checked for plagiarism. If it is copied from some where else, you will fail the assignment.

NOTE: Please upload the archive file in *.zip, *.tar only and report in *.pdf format only to the website https://stavanger.instructure.com/.

Note: The assignment is individual and can **NOT** be solved in groups.

Project Title

Abstract

A one-paragraph summary of the entire assignment consists of any findings, conclusion and recommendations.

1. Introduction

A description of the scientific background for your project, including previous work that your project builds on. (Remember to cite your sources!) The final sentence (analogous to the thesis statement in a term paper) is the objective of your experiment.

2. Design and Implementation

A detailed description (in paragraph format) of the design, procedure, and implementation of your project. This should be the main part of the report.

3. Test Results

Results of testing the software, as you observed/recorded them. Note that this section is only for observations you make during testing. Your analysis belongs in the Discussion section.

4. Discussion

Your analysis of what your testing results mean, and your analysis.

5. Conclusion

A short paragraph that restates the objective from your introduction and relates it to your results and discussion, and describes any future improvements that you would recommend.

References

A bibliography of all of the sources you got information from in your report.