Distributed Ledger-Based Vulnerability Database

Huriye ÖZDEMİR
Department of Computer Engineering , Boğaziçi University
Advisor : Prof. Dr. Fatih ALAGÖZ

Abstract

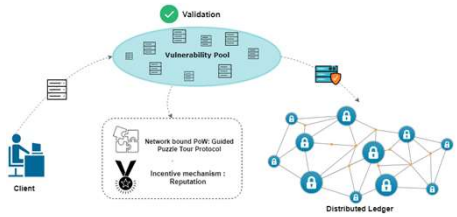
The aim of this project is to create distributed database that enable all users to add vulnerabilities under certain criteria when a vulnerability comes out. The cur-rent centralized database which is named National Vulnerability Database (NVD) is insufficient to include all vulnerabilities and store securely providing all security requirements. There are also many vulnerability database platforms such as exploit-db, OSVDB. As a result, Distributed Ledger based vulnerability database will provide more secure, accessible convenient and aggregated database for all interested parties about information security.

These are some of the advantages that differentiate this project from the cur-rent databases:

- ❖ Decentralization : Eliminating the need for central authority.
- ❖ Utilisable : Providing rights for benefits of every user
- ❖ Aggregation : Collecting all vulnerabilities on a single platform

Introduction

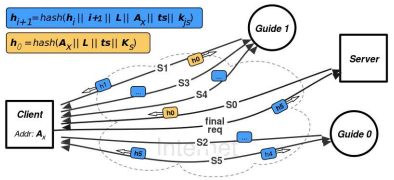
The distributed ledger Database Structure is based on several nodes on a network where each saves a copy of the ledger and have updated data. Each node can construct new transactions and the nodes validates by the consensus algorithm. When the validation of transaction is completed, the ledger updates itself and all nodes have the collected data. The security is accomplished by strong hashing algorithms and cryptographic keys.



Guided Tour Puzzle Protocol

To perform the mining process, which is necessary for the vulnerabilities to be validated and added to the distributed ledger, I used the Guided Tour Puzzle Protocol, a network bound proof of work mechanism. Guided Tour Puzzle Protocol is a cryptographic protocol that aims to overcome the computation of puzzle that is created by the server. The clients are required to complete multiple roud trips in a sequential order.

- Initial server request
- Puzzle solving
- Puzzle verification

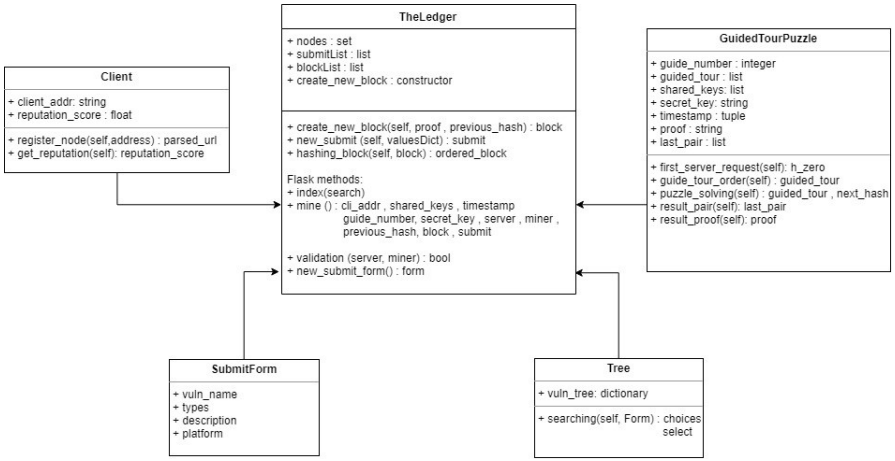


Methodology

I used Python programming language to implement this project. To implement an interface, I used Flask web framework and created web APIs.

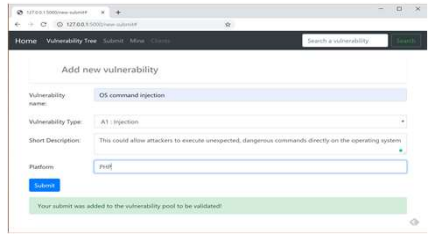
python™ Flask web development, one drop at a time

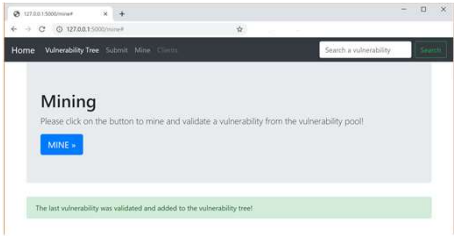
UML Class Diagram



Web APIs

First, the client adds a new vulnerability by filling the required data on the submit tab. Until the new mining process starts, new vulnerabilities are added to a single block with an index number. After submission process, the new vulnerability is added to submit list of the distributed ledger. When the mining process request comes from the miner, guided tour puzzle protocol computes the proof and try to mine the last block in the block list of the ledger.





Data Structure and Classification

After mining process, the vulnerabilities will be added to tree OWASP Top Ten vulnerability classifications. The tree structure is as follows:

A1 : Injection
A2 : Broken Authentication
A3 : Sensitive Data Exposure
A4 : XML External Entities
A5 : Broken Access Control
A6 : Security Misconfiguration
A7 : XSS
A8 : Insecure Deserialization
A9 : Using Components with Known Vulnerabilities
A10: Insufficient Logging & Monitoring

vuln_tree = { "A0": [ ], "A1" : [ ] , "A2": [ ], "A3": [ ] , "A4" : [ ], "A5": [ ], "A6": [ ], "A7" : [ ] , "A8": [ ], "A9": [ ] , "A10": [ ] }

Block Structure :

Conclusion

In this study, I created the distributed ledger based web vulnerability database platform. To eliminate security problems and to provide collective data, this plat-form provide that users can easily access all VDB data, add a new vulnerability and be also miner in the system.

Guided tour puzzle protocol is used to perform proof of work algorithm. It is based on puzzle solving correctly in a sequential order. The new submits from the client are added the blocks and after mining process, vulnerabilities are validated and added to the tree that is classified according to OWASP top Ten web vulnerability document.

Future Work

This study is only created for web application vulnerabilities. The project can be extended by adding different types of vulnerability to make a more comprehensive database. The new required standards can be added for the submission process of the client. The search mechanism may be faster with a strong search algorithm. Rewarding and incentive mechanism can be developed and enhanced for the clients.

Acknowledgements

I would like to thank Prof. Dr. Fatih Alagöz and SATLAB team for their suggestions and comments about the project. I also would like to thank Levent Altay for the support and establishment of the project.