International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 05 Issue: 07 | July 2018 www.irjet.net p-ISSN: 2395-0072

WALLET FOR BITCOIN CRYPTOCURRENCY

Mrs. Deepti Rai[1], Manjesh P Shetty[2], Gautham L Alva[3] , Akshith Hegde[4] , Mohammed Shiran[5]

[1] Assistant Professor, Department of Information Science and Engineering
[2,3,4,5]Students, Department of Information Science and Engineering, Sahyadri College of Engineering and Management, Mangaluru, Karnataka India

---***---

Abstract - *The block chain is a rapidly growing technology that can be adopted in a variety of use cases. The most common application is the famous bit coin crypto currency. The wider this technology spreads, more is the emphasis that has to be placed on the security aspect. Users can interact with the bitcoin network and the transaction block chain using software clients called 'Wallets'. These wallets help a user to send and receive bit coins, and maintain a record of his/her transactions. This project aims to build a hot Wallet for bitcoin storage and transactions in Hierarchical Deterministic structure using BIP0032 and BIP0039 algorithm. The application will be built using C++ for backend and ncurses as front end, wrapped by rust language to enhance the security of the wallet.*

Keyword: Bit coin, Crypt currency, Block chain, Hot wallet, Technology

1. INTRODUCTION

The Block chain is a rapidly growing technology that can be adopted in a variety of use cases. The most common application is the famous Bit coin crypto currency. The wider this technology spreads, the more is the emphasis that has to be placed on the security aspect. Bit coin, unlike most traditional currencies, is a digital crypto currency. Thus, the approach to this kind of currency is completely different, particularly when it comes to acquiring and storing it. Users can interact with the Bit coin network and the transaction block chain using software clients called Wallets. These wallets help a user to send and receive bit coins and maintain a record of his/her transactions. As bit coins are a virtual crypto currency which doesn't have any physical form. The user can restore their balance using private key since there is the private key for every bit coin address of the transaction that is stored in bit coin wallet. Bit coin wallets are developed for many platforms; a desktop version which is compatible with many OS Platforms, the user can also create and access their wallet using web wallet via browsers and also for different mobile OS platforms. This wallet facilitates user for the transaction of bit coin; sending and receiving Bit coins. This project aims to build a Hot Wallet for Bit coin compatible with almost all DISTRO based platforms which is used for storage and Transactions of bit coin. The application will be built using the libbitcoin library based on c++ platform and also JSON programming language is used for backend having ncurses as the User interface for the front end. Even RUST Programming language used in this project as security enhancement and although there are many wallets currently available to manage bit coin storage and Transactions, building a wallet from scratch will help us Experiment with the cryptography behind it, and maybe even use our own combination of cryptographic primitives.

This project conceptually follows hierarchical deterministic wallet structure. This wallet uses BIP 0032 and BIP 0039 Algorithm. HD wallet is able to generate an infinite number of keys from the seed (AKA Master Key) based on BIP 0032, It will generate keys in a hierarchical structure. This seed is technically 128-bit value and in layman terms, it is presented to the user as 12 words mnemonic words, which are nothing but 12 words English words. This seed contains wallet balance (list of private keys) so whenever you restore your wallet using seed (12 mnemonic words) the wallet fetches and restores all the private keys using BIP 0032. The HD wallet has advantage user needs to remember or note down only 12 mnemonic words (seed) with this words user can restore his wallet balance anywhere or any HD wallet. Also, the seed is used after 100,000 rounds of SHA256 which is added advantage to prevent it from cracking the seed. The wallet can generate the infinite number of unique receiving address thus providing the user with greater privacy and making user anonymous. This project is completely open source project and first project having libbitcoin library having the wrapper of RUST language as the security enhancement. It is primarily built for Linux based platform. Users can directly install this software in few easy steps with the help of terminal by cloning the project which is uploaded to GitHub repo, thus successfully running the bit coin wallet on the desktop.

1.1 Architecture Diagram

This architecture illustrates the structure of the HD wallet and its working.

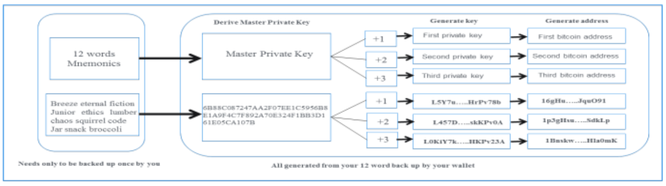


Fig -1: Architecture Diagram for Hot Wallet for Bitcoin

The architecture diagram of the proposed system is shown in Figure 1 the seed generates an infinite number of keys in hierarchical deterministic (HD) Structure. At the initial stage, the HD wallet generates keys from the master key (AKA seed) in tree like structure based on BIP 0032. This seed is also known as 12 mnemonics words in layman terms which are given to the user in order to restore the wallet. 12 mnemonics will restore all the private keys associated with finally it will fetch amount balance of the user restoring your wallet balance and transaction history. The wallet is able to generate an infinite number of address by appending a seed by a counter and is used to derive seemingly in tree-like structure hierarchically and sequentially. When the user restores a wallet using 12 mnemonic words (seed key), the wallet is able to drive all the private keys which are associated with it using BIP 0032. HD wallet produces the tree-like structure of keys without any errors since it uses the one-way SHA-256 hash algorithm. In this architecture, we can see that master key is able to create an infinite tree of cryptographic secrets conceptually in a Hierarchical deterministic structure. The wallet also uses BIP0039 which is bitcoin improvement proposal code for generating mnemonic words.

1.2 Implementation

The term implementation is the process of turning a designed solution into software. It is execution of a project idea, model, system design, system specification, standard, algorithm, or policy. Our Bitcoin Wallet built using the libbitcoin framework, in C++ and JSON. All basic features of a wallet are supported. The HD architecture is followed as per the BIP32 protocol. The wallet connects to libbitcoin servers, which run full nodes, and send and get information from them. The application also makes use of the blockchain.info API to retrieve address balance and transaction history.

- Wallets configuration are JSON files. These JSON files are stored in the local disk soon after creating the wallet.

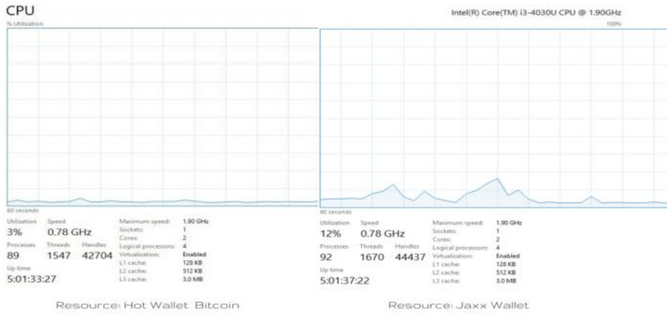
- The system can also import existing wallet by means of providing 12 words mnemonic words where wallet decrypts, fetches and extracts the list of private keys and transaction history balance associated with it.

- Upon sending the bitcoin to wallet asks blockchain.info where the balance of a given address is extracted, and the total balance is calculated given the addresses that are in use, it is achieved using libbitcoin's fetch-utxo to extract the required UTXOs and transaction output indices.

Pseudocode:

```
Create Wallet/Load Wallet:
1. Start
2. if firstTime
Create new wallet
Else
Export existing wallet
3. Stop
Display menu:
1. Start
2. Display private key
3. Generate new receiving address
4. Display balance
5. Display old receiving address
6. Stop
```

2. Results and Discussions



Utilization 3% Speed 0.78 GHz Maximum speed: 1.90 GHz
Sockets: 1
Cores: 2
Processes 89 Threads 1547 Handles 42704 Logical processors: 4
Virtualization: Enabled
L1 cache: 128 KB
Up time 5:01:33:27 L2 cache: 512 KB
L3 cache: 3.0 MB

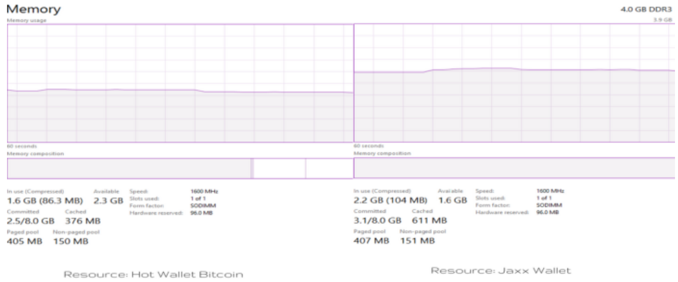
Resource: Hot Wallet Bitcoin

Utilization 12% Speed 0.78 GHz Maximum speed: 1.90 GHz
Sockets: 1
Cores: 2
Processes 92 Threads 1670 Handles 44437 Logical processors: 4
Virtualization: Enabled
L1 cache: 128 KB
Up time 5:01:37:22 L2 cache: 512 KB
L3 cache: 3.0 MB

Resource: Jaxx Wallet

Fig -2: CPU Usage of Hot Wallet for Bitcoin and Jaxx

Figure 2 shows the graphical representation of the CPU usage of hot wallet for bitcoin and Jaxx an existing desktop wallet. According to analyzed survey our wallet uses very less CPU power and it works smoothly on any Linux Distro flavors. From the above graph it can be inferred that the our project uses very less CPU compared to Jaxx Wallet



In use (Compressed) 1.6 GB (86.3 MB) Available 2.3 GB Speed: 1600 MHz
Slots used: 1 of 1
Form factor: SODIMM
Hardware reserved: 96.0 MB
Committed 2.5/8.0 GB Cached 376 MB
Paged pool 405 MB Non-paged pool 150 MB

Resource: Hot Wallet Bitcoin

In use (Compressed) 2.2 GB (104 MB) Available 1.6 GB Speed: 1600 MHz
Slots used: 1 of 1
Form factor: SODIMM
Hardware reserved: 96.0 MB
Committed 3.1/8.0 GB Cached 611 MB
Paged pool 407 MB Non-paged pool 151 MB

Resource: Jaxx Wallet

Fig - 3: Memory Usage of Hot Wallet for Bitcoin and Jaxx Wallet

Figure 3 shows the graphical representation of the memory usage of hot wallet for bitcoin and Jaxx an existing desktop wallet. Front end Ncurses of our project makes use of very low memory compared to other existing wallets From the above graph it can be inferred that the our project uses very less memory compared to Jaxx Wallet.

3. CONCLUSIONS

This project is built using the libbitcoin library in C++ and JSON, currently we have two version: Alpha (Test net) and Beta (Mainnet) version. The front end is built using Ncurses.

The library contains functions to handle the keys, create and broadcast transactions, etc. Future work will support other cryptocurrency to become a multi-wallet and the security can be enhanced using RUST language and enable multi-factor authentication for the wallet.

REFERENCES

[1] Puneet Kumar Kaushal, Dr. Amandeep Bagga and Dr. Rajeev Sobti, "Evolutionnof Bitcoin and Security Risk in Bitcoin Wallets", 2017 International Conference on Computer, Communications and Electronics (Comptelix), Manipal University Jaipur, Malaviya National Institute of Technology Jaipur; IRISWORLD, July 0102, 2017.

[2] Simon Barber, Xavier Boyen, Elaine Shi, and Ersin Uzun "Bitter to better–how to make bitcoin a better currency". In: Keromytis, A.D. (ed.) FC 2012. LNCS, vol.7397, pp. 399–414. Springer, 2012.

[3] Iuon-Chang Lin and Tzu-Chun Liao "International Journal of Network Security", Vol.19, No.5, PP.653-659, Sept-2017.

[4] Rosario Gennaro, Steven Goldfeder and Arvind Narayanan, "Threshold-Optimal DSA/ECDSA Signatures and an Application to Bitcoin Wallet Security", ISBN:9783- 319-39555- 5,PP. 215-512, 2017.

[5] Miraje Gentilal, Paulo Martins and Leonel Sousa, "TrustZone-backed bitcoin wallet",Stockholm, ISBN: 978-1-4503-4869- 0,Sweden — January 2017.

[6] Ilias Giechaskiel, Cremers and Kasper B. Rasmussen, "On Bitcoin Security in the Presence of Broken Cryptographic Primitives", University of Oxford, Oxford UK, ISBN: 978-3-319-45741- 3 ,2017.

[7] Tobias Bamert, Christian Decker, Roger Wattenhofer and Samuel Welten "BlueWallet: The Secure Bitcoin Wallet" published in STM: Security and Trust Management PP.65-80 ISBN: 978-3-319-11851-2,2014.

[8] A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou, "Hawk: The blockchain model of cryptography and privacy-preserving smart contracts," in 2016 IEEE Symposium on Security and Privacy (SP'16) , PP. 839–858, May 2016.

[9] Wei Yin, Qiaoyan Wen and Wenmin Li, "An Anti-Quantum Transaction Authentication Approach in Blockchain", Published in IEEE Access (Volume: 6),PP: 5393– 5401, Electronic ISSN: 2169-3536, 01 January 2018 .

[10] Shuangyu He , Qianhong Wu and Xizhao Luo "A Social-Network-Based Cryptocurrency Wallet-Management Scheme" in IEEE Access (Volume: 6), PP. 7654 - 7663, Electronic ISSN: 2169-3536, 29 January 2018.

[11] Roger S. Pressman, "Software Engineering: A Practitioner's Approach" 7th Edition, McGraw Hill, 2007.

[12] Sommerville, "Software Engineering", 8th Edition, Pearson Education, 2007.

[13] Michael Blaha, James Rumbaugh, "Object-Oriented Modeling and Design with UML", 2nd Edition, Pearson Education, 2005.

[14] Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, Michael Stal,"Pattern-Oriented Software Architecture: A System of Patterns", Volume 1, John Wiley and Sons, 2007.