The 1st International Conference on Innovations in Computer Science & Software Engineering, (ICONICS 2016), December 15-16, 2016

Malleability issues in Block-Chain applications

Kashif Mehboob Khan
Department of Computer Science & Information Technology, NED University of Engineering & Technology-Karachi, Pakistan
kashifmehboob@neduet.edu.pk

Muhammad Mubashir Khan
Department of Computer Science & Information Technology, NED University of Engineering & Technology-Karachi, Pakistan
mmkhan@neduet.edu.pk

***Abstract*— The block-chain mechanism is being implemented in diversified areas of real world applications especially in financial transactions. It has been observed that these transactions may induce malleability in a number of ways to the block chain and do have the tendency to produce the problem of double utilization of same token. Such problems may cause some real time threats to the real world systems if certain conditions are met. For instance, it may cause a genuine owner to be illegally deprived of his assets when a malicious user intentionally discontinues further propagation of his block containing the transaction that moves token from his address to the seller's address. In this paper we propose an attack model to show the double utilization of same token values in block-chain design. The paper concludes by highlighting possible countermeasures of double spending problem.**

Keywords*— *block-chain; malleability; risks; malacious transactions.

I. INTRODUCTION

The block-chain system has been designed for transaction databases (usually financial transactions), which are publicly shared by all the nodes in the network. Every transaction contains a transaction id which is actually hash of all the fields in a transaction. Now it has been known since roughly 2011 that signed transactions are slightly "malleable" in the sense that it is possible to modify a signed transaction in certain minor ways, without invalidating the signature [1][2][11]. Cryptography ensures that the critical details about a transaction can not be changed (like sender, receiver, amount etc.) but certain non-functional fields that do not contribute to the critical parts of a transaction may be changed which causes the hash (transaction id) to be changed for the same transaction.

When transactions in a block-chain are signed, all the data in a transaction is not covered in the creation of transaction hash which makes it possible for an attacker on the block-chain network to change the transaction in such a way that the hash is nullified. This changes the hash of the transaction only, while the output and the message of the transaction remains same. Therefore, in order to avoid transaction malleability one should not accept the transactions that are not mined or confirmed, because all the following transactions in a block-chain depends upon the hashes of the previous transactions, and those hashes can be changed until they are confirmed in a block. Double spending is the possibility to spend a transaction twice or more claiming the same input as a consequence of transaction malleability. One of the transactions will be included in the public ledger while other will be discarded by the network as it will be considered invalid.

One way to check the malleability impact in Bitcoins, is to artificially inject multiple malleable transactions immediately after an original transaction by just changing the nonfunctional fields of a transaction so that new hashes (transaction ID's) may be formed against a practically same transaction. Now if any one of the malleable transactions gets mined first before the original transaction, the miners (computers in the network which validate the transactions) will add this transaction to the block as a valid one because the critical fields in the transaction were unchanged. Now if the sender of the transaction looks for the confirmation of transaction by its transaction id in the transaction database (publicly shared blocks), he is never going to find it as the original transaction (which in our case could not be mined first) would be rejected by the miner as a double spent [2][3]. Malleability may affect badly to other applications based on the data structure of block-chain. For instance, the same person may cast multiple votes.

This paper is organized in three sections. Section 1 introduces the problem and discuss some of the existing approaches which are used to tackle malleability. Section 2 provides references to the work related to the transaction malleability in block chain. In Section 3, an attack model is presented, which is supported by a practical example scenario to show a malleable transaction. The last part of this section shows the experimental working where a block-chain based network model has been formed using multi-chain as a platform. The future work will be based upon applying different methods of injecting malleable transaction in the above mentioned block-chain model and observe its impacts technically and socially on various commonly used applications which can make use of block-chain data structure.

II. RELATED WORK

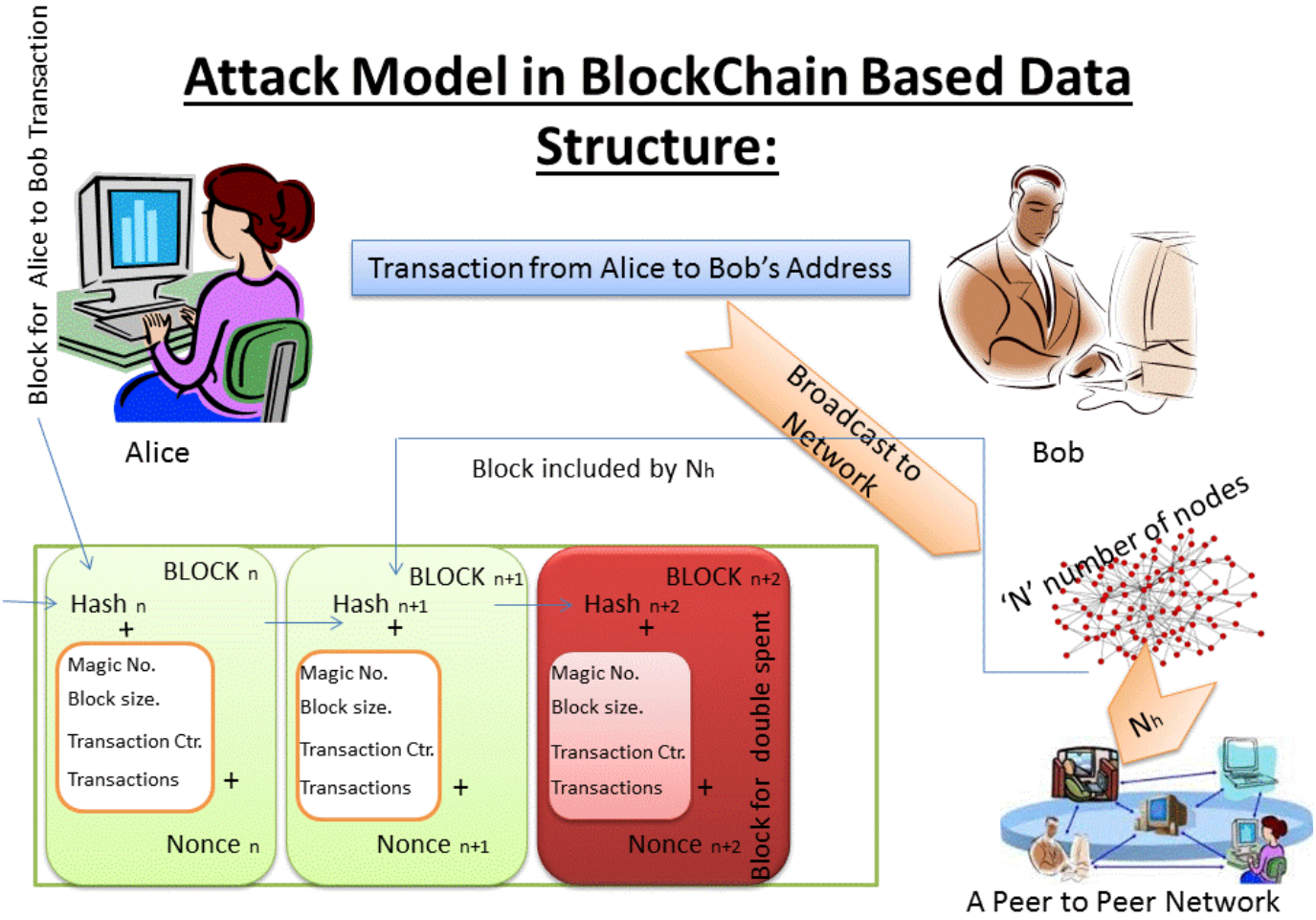
Several approaches have been used out to minimize the risks of double spent transactions which make use of malleability. Among such approaches, one is to wait for certain confirmations which are usually from six miners, to validate a transaction [4]. Even then, it cannot be guaranteed

that the true branch of block chain will be proceeded by true miners as due to network latency, there is a chance that the transaction which has occurred later may be listened earlier than the actual transaction. Therefore, the ordering of events which is based upon time stamps, does not show the true state of the system. There are also some other ways where researchers have made efforts to resolve the problem of double spend, but these are mostly by forcing certain constraints to keep lock of transaction for a particular duration of time until it gets completed. This has been explained by Kadam et al. in their paper [3, 4]. The problem that may arise here is that there is no global time, therefore the area is still open for research community to explore [4].

III. THE ATTACK MODEL

and contains a transaction from sender "S" to receiver "R". A transaction's data structure contains sender's signature, an instruction to send token to receiver's public key, and a hash. This hash is a pointer to a previous transaction output that the sender "S" received and is now spending. That pointer must reference a transaction which was included in some previous block in some consensus chain.

When the receiver finds his transaction included in the consensus block, the sender is acknowledged. Now consider a case in which the next selected node "N" happens to be controlled by the sender "S", then there is a very fair chance that this new node "N" ignores the block that contains the transaction from "S" to "R" in which "S" has moved token from its address to receiver's address and adds new block prior to that block. Moreover, the new proposed block may



In order to conduct test for malleability, an attack model may be developed to operate on the system. Let's assume that a sender 'S' moves his token from his address to receiver "R" on his address against the product or service he wants to utilize. Suppose the transaction successfully occurs from "S" to "R" and it is confirmed by an honest node and gets this transaction into the block chain. So the situation is that there is a block in the block chain which is added by an honest node also contain a transaction that moves values from sender account "S" to another account which is also controlled by "S", thereby generating opportunity to reuse the same token twice [4, 6].

A. *Modelling and Assumptions*

The above mentioned scenario for double spending may be modelled this way. When a double spend attack is made, the

network is in a position that contains a branch which moves the token to the vendor and has *n* blocks extending the one where fork started. In order to model the above mentioned scenario, we can make following assumptions;

i. The system is cryptographically secured and our scope includes how to defend against double utilization of the same token (double spend).

ii. Let the combined hash rate of honest network and the attacker is constant and denoted by *H* and pH belongs to the hash rate of honest network and qH belongs to that of attacker, where p+q=1

iii. Also assume that mining difficulty is constant. *To* is the average time to find a block with a hash rate of H

Let *z* represent the added number of blocks which are created by honest miners from the block where the fork started and has an advantage over miner;

Mathematically,

$$z = n - m$$

where n is the total number of blocks which are created after the transaction that is transferred to vendor, m is the total number $z = n - m$ of blocks which are created by miner (attacker).

The value of z increments or decrements by 1 if the block is added by honest network and miner (attacker) respectively. Here it becomes a continuous–time Markov chain where P/T_o and Q/T_o for increasing and decreasing the chain respectively. Note that if at any stage, the value of *z* becomes negative, it is obvious that the attacker's chain is now bigger and therefore the attack is successful. In order to find whether z will ever be -1, we can take the help of discrete time Markov chain process where the step of the process is defined as the finding of block either by honest network or miner (attacker). P is the probability that the block is found by honest network and *q* be the probability that the block is found by the attacker [2][3][17].

Let a_z shows the probability that the attacker will be succeeded when he is z blocks behind. Now if z is negative then a_z approaches to 100% as he will have a longer branch than the honest miner. Assume that the next block is found by the honest network, which happens with probability p, the attacker will now be $z + 1$ blocks behind and his probability of success will be a_{z+1}. If the next block found will be by the attacker, which happens with probability q, his probability of success will be a_{z-1} [2][14][6].

$$a_z = pa_{z+1} + qa_{z-1}$$

B. Tools & Technologies

Following tools and technologies were considered and explored for creating block-chains and mining of blocks:

- **DESMO-J** is a framework for Discrete-Event Modelling and Simulation which is built on Java. It supports both the process oriented and event oriented modelling style, also known as process interaction approach and event scheduling approach respectively. DESMO-J library can be configured to test and verify the simulation results of discrete events which has been used by many researchers in their research. Obviously, It does not run over real network as it is just a java based library for processing discrete events.

- **Confidence Chains**, a project developed specifically for bitcoin, offers a very flexible way to define the trust relationship that is suitable for a wide range of applications. It also offers a very high degree of irreversibility that does not necessarily depend on the direct authority of one party It has similar anonymity and security characteristics to bitcoins. However, to the best of our knowledge, it was not developed to run over network and incorporate block-chain based applications in general.

- **Multi-Chain** is an off the shelf platform for the creation and deployment of private block-chains either within or between organizations. It aims to overcome a key obstacle to the deployment of block-chain technology in the institutional financial sector by providing the privacy and control required in an easy to use package. Like the bitcoin core software from which it is derived Multi-Chain supports Windows, Linux and Mac servers and provides a simple API's interface and command therefore it was selected as a final platform for building the real network based architecture to perform transactions through block-chain among nodes

C. Implementation of Block-Chain based Network Architecture

The experiment was desired to be conducted by utilizing block-chain in a scenario other than bitcoin to keep focus on block-chain data structure which is the basic engine behind all the transactions and all the issues associated with it. In this example block-chain based scenario has been discussed. Some terminologies necessary to understand the scenario are as follows [21]:

- E-Voting refers here to the block-chain based electronic voting.

- E-Voting Participants include voter, registration authority and candidates.

Following are the steps to build Multi-Chain

i. Configuring Block-Chain for E-Voting

```
>>kashif@kashif-pc:~/.multichain$ mkdir node1
>> kashif@kashif-pc:~/.multichain$ multichain-util
create voteChain
Multichain utilities build 1.0 alpha 16 protocol 1003

Blockchain parameter set was successfully generated.

You can edit it in
/home/kashif/.multichain/voteChain/params.dat before
```

```
running multichaind for the first time.

To generate blockchain please run "multichaind
voteChain".
```

Fig_01

ii. Starting the Block-Chain through root node.

```
>>kashif@kashif-pc:~/.multichain/voteChain$
multichaind voteChain -daemon -
datadir=/home/kashif/.multichain/voteChain


Multichain utilities build 1.0 alpha 16
protocol 1003

Multichain server starting

kashif@kashif-pc:~/.multichain/voteChain$
Looking for genesis block…..
Genisis block found
New users can connect to this node using
multichaind voteChain@192.168.1.107:7721

Node started
```

Fig_02

iii. Creating second node.

```
>>kashif@kashif-pc:~/.multichain$ mkdir
voteChainnode2
>>kashif@kashif-pc:~/.multichain$ multichaind
voteChain @192.168. 1.107:7721 -
datadir=/home/kashif/.multichain/voteChainnode2
-rpcport=6001 -port6002

Multichain utilities build 1.0 alpha 16
protocol 1003

Retrieving blockchain parameters from the seed
node 192.168.1.107:7721 ...

New users can connect to this node using
multichaind voteChain@192.168.1.107:6002

Node started
```

Fig_03

iv. Retrieving chain info.

```
>>voteChain : getinfo {"method" : "getinfo","params"
:[],"id":1,"chain_name"::voteChain"}

{

      "version" : "1.0 alpha 16",
      "protocolversion" : 1003,
      "chainname" : "voteChain",
      "description" : "Blockchain for voting",
      "protocol" ; "multichain",
      "port" : 7721",
       …….. ,
       ……..,
       ……..,

 }
```

Fig_04

v. Generation Of Voter Addresses From Node's Wallet

After creating nodes, multiple addresses were created from each wallet.

```
>>voteChain: getnewaddress
{"method" : "getnewaddress","params"
:[],"id":1,"chain_name"::voteChain"}


1QoVDR6Qkexwsop6f6d204d756c7469436861696eDVk2
```

Fig_05

vi. Issuance of Assets to Addresses

After successful generation of addresses, assets (representing votes) were created and assigned to each address.

```
>>voteChain: issue
1QoVDR6Qkexwsop6f6d204d756c7469436861696eDVk2 vote1 1
1 {"method" : "issue","params" :["
1QoVDR6Qkexwsop6f6d204d756c7469436861696eDVk2"],"id":1
,"chain_name"::voteChain"}


dc05e2c6Qkexwsop6f6d204d756c7469436861abdc6089
{"method" : "listassets","params"
:[],"id":1,"chain_name"::voteChain"}

[

   {

      "name" : "vote1",
      "issuetxid" : "
dc05e2c6Qkexwsop6f6d204d756c7469436861abdc6089",
      "assetref" : "70-265-1500",
      "multiple" : 1,
      "units" : 1,
       "details" : {
       },
       "issueqty" : 1.00000
       :issueraw" : 1

    }

]
```

Fig_06

vii. Performing Transactions between Addresses

A transaction was performed between different nodes.

```
>>voteChain: sendassettoaddress
1HkCKXXXXXXXXXXXXXp7F vote1 1 {"method" : "
sendassettoaddress","params" :["
1HkCKXXXXXXXXXXXXXp7F","vote1"]
,"id":1,"chain name"::voteChain"}
b971da6b60fXXXXXXXXXXXXXXXX43481cf
{
{"method" : "getaddressbalances","params" :["
1HkCKXXXXXXXXXXXXXp7F","vote1"]
,"chain name"::voteChain"}

[
   {
    "name" :  vote1",
     "assetref" : "70-265-1500"
      "qty"  :  1.0000
   }
```

Fig_07

TABLE 1

Platform	Development Technology		
	Client side	***Server side***	***Database***
Ubuntu	Multi-chain Client	Multi-chain Server	Block-chain

IV. CONCLUSION AND FUTURE WORK

Since the block-chaining mechanism has been implemented successfully using Multi-Chain platform and commands using Ubuntu Shell and it has been observed that block-chaining mechanism is not restricted to bitcoin only rather it may be applied on many other diversified application like 'E-voting', therefore the risks of malleability which is mostly associated with bitcoin transactions is in-fact a block-chain oriented problem and needs to be addressed independently of the application in which block-chain based data structure is being utilized. Our future work will target the implementation of above mentioned attack model with reference to E-Voting to observe and test the system over all behavior and its potential impact on the factors associated with the conventional voting system.

ACKNOWLEDGMENT

The research work is supported by N.E.D. University of Engineering & Technology.

REFERENCES

[1] Daniel Kraft, "Difficulty Control for Blockchain-Based Consensus System", Peer-to-Peer Networking and Applications by Springer, March 2015.

[2] M. Rosenfeld. "Analysis of hashrate-based double-spending." [Online]. Available: http://arxiv.org/abs/1402.2009

[3] Mandar Kadam, Praharsh Jha, Shravan Jaiswal, "Double Spending Prevention in Bitcoins Network", International Journal of Computer Engineering and Applications, August 2015.

[4] S. Nakamoto. (2009) Bitcoin: "A peer-to-peer electronic cash system". [Online]. Available: http://bitcoins.info/bitcoin-a-peer-to-peer-electroniccash-system-satoshi-nakamoto

[5] G.O. Karame, E. Androulaki, and S. Capkun. Two bitcoins at the price of one doublespending attacks on fast payments in bitcoin. In Proc. of Conference on Computer and Communication Security, 2012

[6] Arvind Narayanan, Joseph Bonneau, Edward Felten, Andrew Miller, Steven Gold feder, "Bitcoin and Cryptocurrency Technologies", Chapter 2 and 3,Draft Oct0ber 2015

[7] J. Gobel, H.P. Keeler, A.E. Krzesinski, P.G. Taylor, "Bitcoin Blockchain Dynamics: the Selfish-Mine Strategy in the Presence of Propagation Delay", May 2015

[8] Guy Zyskind, Oz Nathan, Alex 'Sandy' Pentland, "Decentralizing Privacy: Using Blockchain to Protect Personal Data" IEEE CS Security and Privacy Workshops, 2015

[9] C. Decker and R. Wattenhofer, "Information propagation in the bitcoin network," in 13th IEEE Conference on Peer-to-Peer Computing, 2013, pp. 1–10

[10] C. Decker and R. Wattenhofer, "Information propagation in the bitcoin network," in 13th IEEE Conference on Peer-to-Peer Computing, 2013, pp. 1–10

[11] L. Bahack, "Theoretical bitcoin attacks with less than half of the computational power." 2013 [Online]. Available: eprint.iacr.org/2013/868

[12] F. Baccelli, I. Norros, and F. Mathieu, "Performance of p2p networks with spatial interactions of peers." [Online]. Available: http://hal.inria.fr/inria-00615523v2

[13] George Foroglou, Anna-Lali Tsilidou, "Further Applications of Blockchain" 12th Student Conference on Managerial Science and Technology, At Athens ,Conference Paper, May 2015

[14] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkitasubramaniam. "l-diversity: Privacy beyond kanonymity". ACM Transactions on Knowledge Discovery from Data (TKDD), 1(1):3, 2007.

[15] Petar Maymounkov and David Mazieres. Kademlia, "A peer-to-peer information system based on the xor metric", In Peer-to-Peer Systems, pages 53–65. Springer, 2002

[16] Arvind Narayanan and Vitaly Shmatikov, "How to break anonymity of the netflix prize dataset".arXiv preprint cs/0610105, 2006.

[17] Latanya Sweeney. k-anonymity, "A model for protecting privacy", International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 10(05):557–570, 2002.

[18] Antonopoulos, A.M, Mastering Bitcoin: Unlocking Digital Cryptocurrencies. O'Reilly Media, 2014

[19] Ittay Eyal and Emin Gun Sirer, "Majority is not Enough: Bitcoin Mining is Vulnerable." ,18th International Conference on Financial Cryptography and Data Security. Barbados, 2014

[20] Swartz, A.: Squaring the Triangle: Secure, Decentralized, Human-Readable Names. January 6th, 2011

[21] http://www.multichain.com/developers/json-rpc-api/