Delft University of Technology

Thesis Project

Double Spending Attack Simulation

by Umeer Mohammad

1 Purpose

The purpose of this document is to describe and reasons regarding the double spending attack that is going to be implemented in the TrustChain simulation built on OMNeT++ (GitHub Repo Link).

2 Simulation overview

In the simulation realized on OMNeT++, there are a set of nodes that spontaneously and randomly do transactions with each other. If a node has a number of coins greater than zero it starts a transaction with a partner, randomly selected in the network. Currently [17-Apr], the network is static, hence during message transfer, there are no lost packet and all the node present in the network are always working and connected.
Every transaction starts from a node that wants to send coins to another one, and it starts with a transaction request message sent to that node. This message contains information about the number of coins and a transaction sequence number. The transaction sequence number is an incremental index used by a node to define a transaction, every transaction has in total two sequence number defined by the parties that create it.

The node that receives the transaction request message will reply back with a chain request message. When received by the first node, it creates a message that contains all the information regarding previous transactions present in its ledger and sends it back.

At this point, the second node can perform a verification by comparison with previously collected pieces of information (via previous transactions of dissemination). If the verification is successful, the transaction will be registered in the chain and an ACK message containing his sequence number will be sent back. Now both of the parties will disseminate information about the new transaction with all its neighbours.

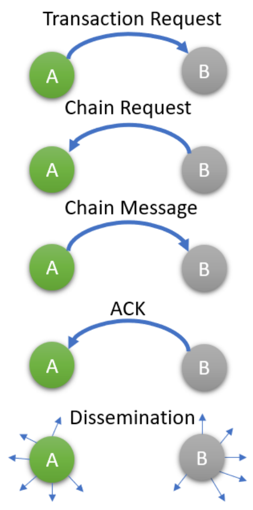

Figure 1: The graphical flow of messages during a transaction

3 Definition of double spending

Double-spending is a problem in which the same digital currency can be spent more than once. In other words, double-spending is an instance in which a transaction uses the same input as another transaction that has already been broadcast on the network.
In the specific case of this simulation, a double spending event can be defined as follow, when a node having γ coins spend an ω amount (where $\omega \geq$1 and $\omega \leq \gamma$) in a transaction with another subject but at the end after the trade, the total amount of coin own by the node remain equals to γ, which will be spent somewhere else.

4 Detection method

A node can detect double spending during two moments, the first one more direct is during the verification phase. And it can be identified if and only if the node has previously collected information regarding the malicious node transactions. This data can be collected via a previous transaction with the evil node itself, or more likely as a result of receiving dissemination messages from the network.
During the verification phase, a node has to compare the received information with previously collected knowledge. There are two main conditions that can lead to a failure of the verification:

- The node performing the verification has information about a transaction in which the partner was involved, but there is no information about that event in the chain sent by him. (Block Omission)
- The local knowledge regarding the partner transaction differs from what he is claiming in the chain. (Block Overwriting)

The second way to detect an evil node is by analyzing the dissemination information. During this phase, the network is broadcasted with the ids, sequence numbers and transaction value of a transaction. Hence, by comparing these ids and sequence numbers with previously collected knowledge, it is possible to detect reuse of sequence numbers, which is an unremarkable trace of a double spending event.

5 Variety of attacks

The way to detect an attack is unique and well definable, but the dynamic with which this can be performed can vary a lot, and consequently with that the detection time also changes.
Here are some examples of attack strategies that an evil node can perform.

- Do one malicious transaction and then a correct one. In this case, after completing the first transaction, the evil node does not save the details about it and starts the second one.
- Do one malicious transaction and then a correct one, but do not disseminate it at the end.
- Do one malicious transaction and then many normal in row.
- Do multiple malicious transactions in a row.
- Just do a malicious and go offline.

The best way for an evil node to increase the possibility of not being detected is to limit the information dissemination at the end of the malicious transaction and so stop performing transactions with any other node.
However, a double spending attack can be considered successful only if the evil node performs a second transaction where he spends a number of coins that he is not supposed to have. Hence for all the above reason i believe that the best attack strategy is the second one in the list.

Because of the fact that information dissemination takes time to cover the entire network an evil node can exploit this characteristic, and conduct transactions with nodes that are far away from each other. By doing this the two transactions dissemination will take longer to reach the opposite node's area, resulting in longer longevity for the evil node before being detected.

In the case or transactions initiated by other nodes in the interest of the evil node, there are two strategies that can be performed. The evil node can simply ignore the request or engage in the transaction. The best option is clearly the first one because limits the exposure of the evil node's chains information to the network.

6 Detection time influences

The double spending detection time is strongly depended on the dissemination of the information in the network. Hence any changes in the factors that affect it will also result in variations of the detection time.

- **Distance between attacked nodes**
  As described before, the distance in terms of the network hop between the attacked nodes can affect the detection time. The bigger the distance is the longer will take for the nodes in between to receive the dissemination of the transactions, hence it will take longer before the evil node will be detected by them.
- **Network transmission speed and congestion**
  These parameters affect all the messages in the network and so the dissemination speed. Having a slow network transmission speed will result in delays for the dissemination messages, and in case of heavily congested nodes, these messages can get clogged in log transmission queues.
- **Network stability**
  Having a mutable network where nodes can go offline, can affect the distance between the attacked nodes and/or create circumstances where there are congested network paths. And as we saw in the previous point these factors can affect the detection time.
- **Transaction starting time**
  The time between the two evil transactions is also an important factor. If time is long enough the network will be already saturated from the first transaction dissemination. Consequently, the second evil transaction will be caught directly in the chain verification phase, this because, the attacked node would have by now already updated his local knowledge with the dissemination information of the first transaction.