1 Abstract

We propose a decentralized reputation system that can replace the word-of-mouth, stars- and review-based systems. The basic idea is that a member A trusts her friends with a certain value (with a 1/2 multisig), thus risking to lose their value. When A wants to transfer value V to a (maybe previously unknown) member B, A asks the system if she trusts B enough to transfer this value to B. The system will search throughout the network for trust paths that begin from A and reach B and add up to V and will answer whether the proposed transaction is within the trust capabilities of A towards B. If the answer is positive, it means that transferring value V to B will not raise the risk for A to lose their value. Note: we use Bitcoin terminology.

2 Introduction

3 Tags/Keywords

decentralized, trust, web-of-trust, bitcoin, multisig, line-of-credit, trust-as-risk, flow

4 Related Work

5 Key points

6 Definitions

Definition 6.1 (Direct Trust from A to B, $DTr_{A \to B}$)**.**
Total amount of value that exists in 1/{A,B} multisigs in the utxo, where the money is deposited by A.

Definition 6.2 (B steals x from A)**.**
B steals value x from A when B reduces the $DTr_{A \to B}$ by x. This makes sense when $x \leq DTr_{A \to B}$.

Definition 6.3 (Honest strategy)**.**
A member A is said to follow the honest strategy if for any value x that is stolen from her, she substitutes it by stealing from others that trust her value equal to $min(x, \sum_{B \in members} DTr_{B \to A})$ and she takes no other action.

Definition 6.4 (Indirect trust from A to B $Tr_{A \to B}$)**.**
Value that A will lose if B steals the maximum amount she can steal (all her incoming trust) and everyone else follows the honest strategy.

7 Theorems-Algorithms

Theorem 7.1. $Tr_{A \to B} = MaxFlow_{A \to B}$ *(Treating trusts as capacities)*

Proof.

1. $Tr_{A \to B} \geq MaxFlow_{A \to B}$ because by the definition of $Tr_{A \to B}$, B leaves taking with him all the incoming trust, so there is no trust flowing towards him after leaving. $Tr_{A \to B} < MaxFlow_{A \to B}$ would imply that after B left, there would still remain trust flowing from A to B.

2. $Tr_{A \to B} \leq MaxFlow_{A \to B}$
   Suppose that $Tr_{A \to B} > MaxFlow_{A \to B}$ (1). Then, using the min cut - max flow theorem we see that there is a set of capacities $C = \{c_1, ..., c_n\}$ with flows $X = \{x_1, ..., x_n\}$ such that $\sum_{i=1}^{n} x_i = MaxFlow_{A \to B}$ and, if severed ($c_i' = 0 \; \forall i \in \{1, ..., n\}$) the flow from A to B would be 0, or, put differently, there would be no directed trust path from A to B. No strategy followed by B could reduce the value of A, so our supposition (1) cannot be true.

Combining the two results, we see that $Tr_{A \to B} = MaxFlow_{A \to B}$. □

Theorem 7.2. *If everybody follows the honest strategy, nobody steals any amount from anybody.*

Proof. According to the definition of the honest strategy, a member steals a value only when he is stolen at least the same value. Let A be a member of the network. Suppose that A steals value V from member B. Since A follows the honest strategy, she has been stolen at least V from another member, C. The same argument holds for C. This reasoning cannot be repeated *ad infinitum* because the network has finite members and finite total value. Thus member A could not have stolen any value. ☐

Theorem 7.3 (Trust transfer theorem (flow terminology))**.**
Let s source, t sink,
$X_s = \{x_{s,1}, ..., x_{s,n}\}$ *outgoing flows from s,*
$X_t = \{x_{1,t}, ..., x_{m,t}\}$ *incoming flows to t,*
$U_s = \{u_{s,1}, ..., u_{s,n}\}$ *outgoing capacities from s,*
$U_t = \{u_{1,t}, ..., u_{m,t}\}$ *incoming capacities to t,*
V the value to be transferred.
Nodes apart from s, t cannot create or consume flow.
Obviously $maxFlow = F = \sum_{i=1}^{n} x_{t\to i}$.

```
    /                ....               \
   / x_s1/u_s1                x_1t/u_1t   \
   /                                      \
  /                                        \
 / x_s2/u_s2                    x_2t/u_2t  \
s-------------       ....       ------------t
 \     .                                .   /
  \    .                                .  /
   \   .                                . /
    \ x_sn/u_sn       ....     x_mt/u_mt  /
    \                                   /
```

We create a new graph where

1. $\sum_i u'_{s,i} = F - V$
2. $u'_{s,i} \leq x_{s,i} \forall i \in \{1, ..., n\}$

It holds that $maxFlow' = F' = F - V$.

Proof.

1. It is impossible to have $F' > F - V$ because $F' \leq \sum u'_{s,i} = F - V$.
2. It is impossible to have $F' < F - V$.
   Let i be a node such that $x_{s,i} > 0$ and $I = \{(i, j) \in E\}$ the set of direct trusts outgoing from i. In the initial graph we have $x_{s,i} = \sum_j x_{i,j}, F = \sum_i x_{s,i}$ and in the new graph we have $x'_{s,i} = u'_{s,i} \leq x_{s,i}, F' = \sum_i x'_{s,i}, x_{i,j} \leq u_{i,j} = u'_{i,j} \; \forall \, j, i$. We can construct a set $X'_i = \{x'_{i,j}\}$ of flows such that $x'_{i,j} \leq x_{i,j}$ and $\sum_j x'_{i,j} = x'_{s,i}$. This shows that there is a possible flow such that $F' = F - V$, so the maxFlow algorithm will not return a flow less than $F - V$.
   Example construction:
   $x'_{i,j} = x_{i,j} \; \forall \, j \in \{1, ..., k\}$ with k such that

   (a) $\sum_{j=1}^{k} x_{i,j} \leq x'_{s,i}$ and

   (b) $\sum_{j=1}^{k+1} x_{i,j} > x'_{s,i}$

   $x'_{i,(k+1)} = x'_{s,i} - \sum_{j=1}^{k} x'_{i,j}$
   $x'_{i,j} = 0 \; \forall \, j \in \{k+2, ..., |X'_i|\}$

☐

Corollary 7.1 (Requirement for $\sum_i u'_{s,i} = F - V$, $u'_{s,i} \leq x_{s,i}$)**.**
In the setting of 7.3, it is impossible to have $maxFlow' = F - V$ *if* $\sum_i u'_{s,i} > F - V \wedge u'_{s,i} \leq x_{s,i} \forall i \in \{1, ..., n\}$.

Proof. Due to 7.3, $maxFlow' = F - V$ if $\sum_i u'_{s,i} = F - V \wedge u'_{s,i} \leq x_{s,i} \; \forall \; i \in \{1, ..., n\}$. If we create new capacities such that $u''_{s,i} \leq x_{s,i} \; \forall \; i \in \{1, ..., n\}$, then obviously $maxFlow'' = \sum_i u''_{s,i}$. If additionally $\sum_i u''_{s,i} > F - V$, then $maxFlow'' > F - V$. □

Here we show three naive algorithms for calculating new direct trusts so as to maintain invariable risk when paying a trusted party.

Algorithm 1: First-come, first-served trust transfer

```
   Input  : $x_i$ flows, $n$ flows number, $V$ value
   Output: $u'_i$ capacities
 1 $F \leftarrow \sum_{i=1}^{n} x_i$
 2 if $F < V$ then
 3     return error
 4 $Fcur \leftarrow F$
 5 for $i \leftarrow 1$ to $n$ do
 6     $u'_i \leftarrow x_i$
 7 $j \leftarrow 1$
 8 while $Fcur > F - V$ do
 9     $reduce \leftarrow min(u'_j, Fcur - V)$
10     $Fcur \leftarrow Fcur - reduce$
11     $u'_j \leftarrow u'_j - reduce$
12     $j \leftarrow j + 1$
```

Algorithm 2: Absolute equality trust transfer TODO

```
   Input  : $x_i$ flows, $n$ flows number, $V$ value
   Output: $u'_i$ capacities
 1 $F \leftarrow \sum_{i=1}^{n} x_i$
 2 if $F < V$ then
 3     return error
 4 for $i \leftarrow 1$ to $n$ do
 5     $u'_i \leftarrow x_i$
 6 while $\sum_{i=1}^{n} u'_i > F - V$ do
```

Algorithm 3: Proportional equality trust transfer

```
   Input  : $x_i$ flows, $n$ flows number, $V$ value
   Output: $u'_i$ capacities
 1 $F \leftarrow \sum_{i=1}^{n} x_i$
 2 if $F < V$ then
 3     return error
 4 for $i \leftarrow 1$ to $n$ do
 5     $u'_i \leftarrow x_i - \frac{F-V}{\sum_{i=1}^{n} x_i} \cdot x_i$
 6 while $\sum_{i=1}^{n} u'_i > F - V$ do
```

Proof of correctness. In all three algorithms, we have $u'_i <= x_i$ because in the only case where u'_i is altered after its initialisation, it is reduced. Furthermore, a total of V is subtracted from all the u'_i, thus $\sum_{i=1}^{n} u'_i = F - V$. □

However, we need to minimize $\sum_{i=1}^{n} (u_i - u'_i)$.

8 Further Research

9 References