

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, $DTr_{A \rightarrow B}$).

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 $DTr_{A \rightarrow B}$ by x . This makes sense when $x \leq DTr_{A \rightarrow B}$.

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 $\min(x, \sum_{B \in \text{members}} DTr_{B \rightarrow A})$ and she takes no other action.

Definition 6.4 (Indirect trust from A to B $Tr_{A \rightarrow B}$).

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. $Tr_{A \rightarrow B} = MaxFlow_{A \rightarrow B}$ (Treating trusts as capacities)

Proof.

1. $Tr_{A \rightarrow B} \geq MaxFlow_{A \rightarrow B}$ because by the definition of $Tr_{A \rightarrow B}$, B leaves taking with him all the incoming trust, so there is no trust flowing towards him after leaving. $Tr_{A \rightarrow B} < MaxFlow_{A \rightarrow B}$ would imply that after B left, there would still remain trust flowing from A to B.
2. $Tr_{A \rightarrow B} \leq MaxFlow_{A \rightarrow B}$
Suppose that $Tr_{A \rightarrow B} > MaxFlow_{A \rightarrow B}$ (1). Then, using the min cut - max flow theorem we see that there is a set of capacities $C = \{c_1, \dots, c_n\}$ with flows $X = \{x_1, \dots, x_n\}$ such that $\sum_{i=1}^n x_i = MaxFlow_{A \rightarrow B}$ and, if severed ($c'_i = 0 \forall i \in \{1, \dots, n\}$) 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 $Tr_{A \rightarrow B} = MaxFlow_{A \rightarrow B}$. □

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. \square

Theorem 7.3 (Trust transfer theorem (flow terminology)).

Let s source, t sink,

$X_s = \{x_{s,1}, \dots, x_{s,n}\}$ outgoing flows from s ,

$X_t = \{x_{1,t}, \dots, x_{m,t}\}$ incoming flows to t ,

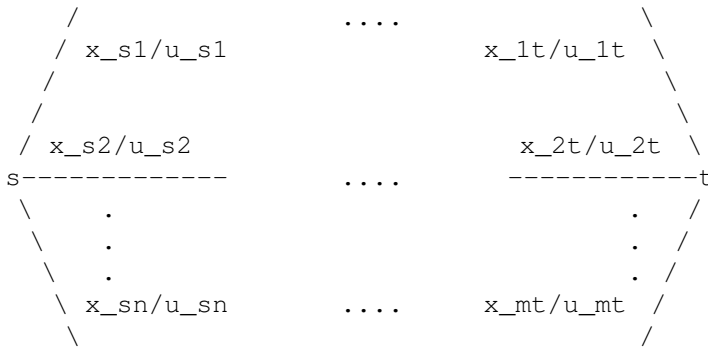
$U_s = \{u_{s,1}, \dots, u_{s,n}\}$ outgoing capacities from s ,

$U_t = \{u_{1,t}, \dots, u_{m,t}\}$ incoming capacities to t ,

V the value to be transferred.

Nodes apart from s, t cannot create or consume flow.

Obviously $\maxFlow = F = \sum_{i=1}^n x_{t \rightarrow i}$.



We create a new graph where

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

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:

$x'_{i,j} = x_{i,j} \forall j \in \{1, \dots, k\}$ with k such that

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

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

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

$$x'_{i,j} = 0 \forall j \in \{k+2, \dots, |X'_i|\}$$

\square

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

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

Theorem 7.4 (Trust-saving Theorem).

$$u'_i = F_{A_i \rightarrow B} \Leftrightarrow u'_i = u_i \forall i$$

Proof. We know that $x_i \leq F_{A_i \rightarrow B}$, thus we can see that any increase in u'_i beyond $F_{A_i \rightarrow B}$ will not influence x_i and subsequently will not incur any change on the rest of the flows. \square

Theorem 7.5 (Invariable trust reduction with naive algorithms).

If $u'_i \leq x_i \forall i$, Trust Reduction (TrR) invariable \forall configurations of x_i

Proof. $TrR = \sum_{i=1}^n TrR_i$ total Trust Reduction, $TrR_i = u_i - u'_i$, Trust Reduction on i. Since $u'_i \leq x_i \forall i$ it is $x'_i = u'_i$, thus $TrR_i = u_i - x'_i$. We know that $\sum_{i=1}^n x'_i = F - V$, so we have $TrR = \sum_{i=1}^n TrR_i = \sum_{i=1}^n (u_i - x'_i) = \sum_{i=1}^n u_i - F + V$ independent of x'_i, u'_i \square

Theorem 7.6 (Dependence impossibility theorem).

$$\frac{dx_j}{dx_i} = 0 \text{ with } x_i \text{ the flow from MaxFlow} \Rightarrow \frac{dx_j}{dx_i} = 0 \forall x'_i < x_i \text{ ceteris paribus}$$

Proof. TODO \square

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  $F_{cur} \leftarrow F$ 
5 for  $i \leftarrow 1$  to  $n$  do
6   |  $u'_i \leftarrow x_i$ 
7  $i \leftarrow 1$ 
8 while  $F_{cur} > F - V$  do
9   |  $reduce \leftarrow \min(u'_i, F_{cur} - V)$ 
10  |  $F_{cur} \leftarrow F_{cur} - reduce$ 
11  |  $u'_i \leftarrow u'_i - reduce$ 
12  |  $i \leftarrow i + 1$ 

```

Algorithm 2: Absolute 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   |  $u'_i \leftarrow x_i$ 
6  $reduce \leftarrow \frac{V}{n}$ 
7  $reduction \leftarrow 0$ 
8  $empty \leftarrow 0$ 
9  $i \leftarrow 0$ 
10 while  $reduction < V$  do
11   | if  $u'_i > 0$  AND  $x_i < reduce$  then
12     |  $empty \leftarrow empty + 1$ 
13     |  $reduce = reduce + \frac{u'_i}{n - empty}$ 
14     |  $reduction \leftarrow reduction + u'_i$ 
15     |  $u'_i \leftarrow 0$ 
16   | else if  $x_i \geq reduce$  then
17     |  $reduction \leftarrow reduction + u'_i - (x_i - reduce)$ 
18     |  $u'_i \leftarrow x_i - reduce$ 
19   |  $i \leftarrow (i + 1) \bmod n$ 

```

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   |  $u'_i \leftarrow x_i - \frac{F-V}{\sum_{i=1}^n x_i} \cdot x_i$ 

```

Proof of correctness. In all three algorithms, we have $u'_i \leq x_i$ 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 $\sum_{i=1}^n u'_i = F - V$. \square

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

8 Further Research

9 References