

1 Abstract

Reputation in centralized systems typically uses stars and review-based trust. These systems require extensive manual intervention and secrecy to avoid manipulation. In decentralized systems this luxury is not available as the reputation system should be autonomous and open source. Previous peer-to-peer reputation systems define trust abstractly and do not allow for financial arguments pertaining to reputation. We propose a concrete sybil-resilient decentralized reputation system in which direct trust is defined as lines-of-credit using bitcoin's 1-of-2 multisig. We introduce a new model for bitcoin wallets in which user coins are split among trusted friends. Indirect trust is subsequently defined using a transitive property. This enables formal game theoretic arguments pertaining to risk analysis. Using our reputation model, we prove that risk and max flows are equivalent and propose several algorithms for the redistribution of trust so that a decision can be made on whether an anonymous third party can be indirectly trusted. In such a setting, the risk incurred by making a purchase from an anonymous vendor remains invariant. Finally, we prove the correctness of our algorithms and provide optimality arguments for various norms.

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 (Members).

The set $\mathcal{M} = V(G)$ is the set of all members in the network, otherwise understood as the set of all pseudonymous identities.

Definition 6.2 ((In/Out) Neighbourhood of A , $N^+(A)$, $N^-(A)$, $N(A)$).

1. Let $N^+(A)$ be the set of members B that A directly trusts with any positive value. More formally, $N^+(A) = \{B \in \mathcal{M} : DTr_{A \rightarrow B} > 0\}$. $N^+(A)$ is called out neighbourhood of A .
2. Let $N^-(A)$ be the set of members B that directly trust A with any positive value. More formally, $N^-(A) = \{B \in \mathcal{M} : DTr_{B \rightarrow A} > 0\}$. $N^-(A)$ is called in neighbourhood of A .
3. Let $N(A)$ be the set of members B that either directly trust or are directly trusted by A with any positive value. More formally, $N(A) = N^+(A) \cup N^-(A)$. $N(A)$ is called neighbourhood of A .
4. Let $N(A)_i$ (respectively $N^+(A)_i$, $N^-(A)_i$) be the i -th element of set $N(A)$ (respectively of $N^+(A)$, $N^-(A)$), according to an arbitrary but constant enumeration of the set members.

Definition 6.3 (Direct Trust from A to B , $DTr_{A \rightarrow B}$).

Total amount of value that exists in 1-of- A, B multisigs in the utxo, where the money is deposited by A .

Definition 6.4 (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 6.5 (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 $\min(x, \sum_{B \in \mathcal{M}} DTr_{B \rightarrow A})$ and she takes no other action.

Definition 6.6 (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.

Definition 6.7 (Trust Reduction).

Let $A, B \in \mathcal{M}$, x_i flow to $N^+(A)_i$ resulting from $maxFlow(A, B)$, u_i current $DTr_{A \rightarrow N^+(A)_i}$, u'_i new $DTr_{A \rightarrow N^+(A)_i}$, $i \in \{1, \dots, |N^+(A)|\}$.

1. The Trust Reduction on neighbour i , δ_i is defined as $\delta_i = u_i - u'_i$.
2. The Flow Reduction on neighbour i , Δ_i is defined as $\Delta_i = x_i - u'_i$.

We will also use the standard notation for 1-norm and ∞ -norm, that is:

1. $\|\delta_i\|_1 = \sum_{i \in N^+(A)} \delta_i$
2. $\|\delta_i\|_\infty = \max_{i \in N^+(A)} \delta_i$.

Definition 6.8 (Restricted Flow).

Let $A, B \in \mathcal{M}$, $i \in \{1, \dots, |N^+(A)|\}$.

1. Let $F_{A \rightarrow B}$ be the flow from A to $N^+(A)_i$ as calculated by the $maxFlow(A, B)$ (x'_i) when $u'_i = u_i$, $u'_j = 0 \forall j \in \{1, \dots, |N^+(A)|\} \wedge j \neq i$.
2. Let $S \subset N^+(A)$. Let $F_{A \rightarrow B}$ be the sum of flows from A to S as calculated by the $maxFlow(A, B)$ ($\sum_{i=1}^{|S|} x'_i$) when $u'_C = u_C \forall C \in S$, $u'_D = 0 \forall D \in N^+(A) \setminus S$.

7 Theorems-Algorithms

Theorem 7.1 (Saturation theorem).

Let s source, $n = |N^+(s)|$, $x_i, i \in \{1, \dots, n\}$, flows to s 's neighbours as calculated by the $maxFlow$ algorithm, u'_i new direct trusts to the n neighbours and x'_i new flows to the neighbours as calculated by the $maxFlow$ algorithm with the new direct trusts, u'_i . It holds that $\forall i \in \{1, \dots, n\}, u'_i \leq x_i \Rightarrow x'_i = u'_i$.

Proof.

1. $\forall i \in \{1, \dots, n\}, x'_i > u'_i$ is impossible because a flow cannot be higher than its corresponding capacity. Thus $\forall i \in \{1, \dots, n\}, x'_i \leq u'_i$.
2. In the initial configuration of u_i and according to the flow problem setting, a combination of flows y_i such that $\forall i \in \{1, \dots, n\}, y_i = u'_i$ is a valid, albeit not necessarily maximum, configuration with a flow $\sum_{i=1}^n y_i$. Suppose that $\exists j \in \{1, \dots, n\} : x'_j < u'_j$ as calculated by the $maxFlow$ algorithm with the new direct trusts, u'_i . Then for the new $maxFlow$ F' it holds that $F' = \sum_{i=1}^n x'_i < \sum_{i=1}^n y_i$ since $x'_j < y_j$ which is impossible because the configuration $\forall i \in \{1, \dots, n\}, x'_i = y_i$ is valid since $\forall i \in \{1, \dots, n\}, y_i = u'_i$ and also has a higher flow, thus the $maxFlow$ algorithm will prefer the configuration with the higher flow. Thus we deduce that $\forall i \in \{1, \dots, n\}, x'_i \geq u'_i$.

From (1) and (2) we conclude that $\forall i \in \{1, \dots, n\}, x'_i = u'_i$. □

Theorem 7.2 (Trust flow theorem - TOCHECK).

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

Proof.

1. We will show that $Tr_{A \rightarrow B} \leq MaxFlow_{A \rightarrow B}$. We know that $MaxFlow_{A \rightarrow B} = MinCut_{A \rightarrow B}$. We will show that, if everybody except A and B follows the honest strategy, $Tr_{A \rightarrow B} \leq MinCut_{A \rightarrow B}$. Suppose that in round i all the members of the MinCut, P , have stolen the maximum value they can from members that belong in the MaxFlow graph and nobody in the partition in which A belongs has stolen yet any value. Let the total stolen value from the MinCut members be St . It is obvious that $St_i \leq MinCut_{A \rightarrow B}$, because otherwise there would exist $u \in P$ that doesn't follow the honest strategy, since they stole more than they were stolen from. The same argument holds for any round $i' > i$ because in each round an honest player can steal only up to the value she has been stolen. It is also impossible that the St increase further due to stolen value from members of the partition of B since members of P disconnect the two partitions and have already played their turns, thus $\forall i' > i, St_{i'} \leq St_i$. There exists a round, k , when all the honest players stop stealing, so in the worst case A will have been stolen $Tr_{A \rightarrow B} = St_k \leq MinCut_{A \rightarrow B} = MaxFlow_{A \rightarrow B}$.
2. We can see that $Tr_{A \rightarrow B} \geq MaxFlow_{A \rightarrow B}$ because the strategy where each one of the non-idle players steals value equal to the incoming flows from their respective friends is a valid strategy that does not contradict with the honest strategy, since for every honest player w it holds that $\sum_{v \in N^-(w)} x_{vw} = \sum_{v \in N^+(w)} x_{wv}$ and according to the strategy each honest player will have been stolen value equal to $\sum_{v \in N^+(w)} x_{wv}$.

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

Theorem 7.3 (Honest world theorem).

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

Proof. Suppose that there exists a series of stealing actions represented by a vector where $action_i =$ "member i steals value $V > 0$ from member $i+1$ ". This vector must have an initial element, $action_1$. However, member 1 follows the honest strategy, thus somebody must have stolen from her as well, so member 1 cannot be the initial element. We have a contradiction, thus there cannot exist a series of stealing actions when everybody is honest. \square

Theorem 7.4 (Trust transfer theorem (flow terminology) - TOCHECK).

Let s source, t sink, $n = N^+(s)$

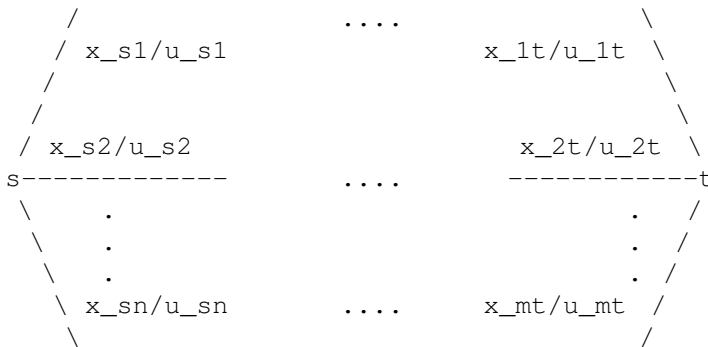
$X_s = \{x_{s,1}, \dots, x_{s,n}\}$ outgoing flows from s ,

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

V the value to be transferred.

Nodes apart from s, t follow the honest strategy.

Obviously $maxFlow = F = \sum_{i=1}^n x_{s,i}$.



We create a new graph where

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

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

Proof. From theorem 7.1 we can see that $x'_i = u'_i$. It holds that $F' = \sum_{i=1}^n x'_i = \sum_{i=1}^n u'_i = F - V$. \square

Corollary 7.1 (Requirement for $\sum_{i=1}^n u'_{s,i} = F - V$, $u'_{s,i} \leq x_{s,i}$).

In the setting of 7.4, it is impossible to have $\maxFlow' = F - V$ if $\sum_{i=1}^n u'_{s,i} > F - V \wedge \forall i \in \{1, \dots, n\}, u'_{s,i} \leq x_{s,i}$.

Proof. Due to 7.4, $\maxFlow' = F - V$ if $\sum_{i=1}^n u'_{s,i} = F - V \wedge \forall i \in \{1, \dots, n\}, u'_{s,i} \leq x_{s,i}$. If we create new capacities such that $\forall i \in \{1, \dots, n\}, u''_{s,i} \leq x_{s,i}$, then obviously $\maxFlow'' = \sum_{i=1}^n u''_{s,i}$. If additionally $\sum_{i=1}^n u''_{s,i} > F - V$, then $\maxFlow'' > F - V$. \square

Theorem 7.5 (Trust-saving Theorem).

$$\forall i \in \{1, \dots, |N^+(A)|\}, u'_i = F_{A_i \rightarrow B} \Leftrightarrow u'_i = u_i$$

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 7.6 (Invariable trust reduction with naive algorithms).

Let A source, $n = |N^+(A)|$ and u'_i new direct trusts. If $\forall i \in \{1, \dots, n\}, u'_i \leq x_i$, Trust Reduction $\|\delta_i\|_1$ is independent of x_i, u'_i
 \forall configurations of x_i

Proof. Since $\forall i \in \{1, \dots, n\}, u'_i \leq x_i$ it is (according to 7.1) $x'_i = u'_i$, thus $\delta_i = u_i - x'_i$. We know that $\sum_{i=1}^n x'_i = F - V$, so we have $\|\delta_i\|_1 = \sum_{i=1}^n \delta_i = \sum_{i=1}^n (u_i - x'_i) = \sum_{i=1}^n u_i - F + V$ independent of x'_i, u'_i \square

Theorem 7.7 (Dependence impossibility theorem).

$$\frac{\partial x_j}{\partial x_i} = 0 \text{ with } x_i \text{ the flow from MaxFlow} \Rightarrow \forall x'_i \leq x_i, \frac{\partial x_j}{\partial x_i} = 0 \text{ ceteris paribus}$$

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, $n = |N^+(s)|$, V value
Output: u'_i capacities

```

1  $F \leftarrow \sum_{i=1}^n x_i$ 
2 if  $F < V$  then
3   | return  $\perp$ 
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  |  $i \leftarrow i + 1$ 
13 return  $U' = \bigcup_{i=1}^n \{u'_i\}$ 

```

Algorithm 2: Absolute equality trust transfer($\|\Delta_i\|_\infty$ minimizer)

Input : x_i flows, $n = |N^+(s)|$, V value
Output: u'_i capacities

```

1  $F \leftarrow \sum_{i=1}^n x_i$ 
2 if  $F < V$  then
3   | return  $\perp$ 
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  $u'_i > 0 \wedge x_i < reduce$  then
12     |  $empty \leftarrow empty + 1$ 
13     |  $reduce = reduce + \frac{x_i - reduce - u'_i}{n - empty}$ 
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   |  $i \leftarrow (i + 1) \bmod n$ 
20 return  $U' = \bigcup_{i=1}^n \{u'_i\}$ 

```

Algorithm 3: Proportional equality trust transfer

Input : x_i flows, $n = |N^+(s)|$, V value
Output: u'_i capacities

```

1  $F \leftarrow \sum_{i=1}^n x_i$ 
2 if  $F < V$  then
3   | return  $\perp$ 
4 for  $i \leftarrow 1$  to  $n$  do
5   |  $u'_i \leftarrow x_i - \frac{V}{F} x_i$ 
6 return  $U' = \bigcup_{i=1}^n \{u'_i\}$ 

```

Algorithm 4: $\|\delta_i\|_\infty$ minimizer

Input : $X = \{x_i\}$ flows, $n = |N^+(s)|$, V value
Output: u'_i capacities

```

1  $F \leftarrow \sum_{i=1}^n x_i$ 
2  $\delta \leftarrow \text{BinSearch}(0, V, F, n, X)$ 
3 for  $i \leftarrow 1$  to  $n$  do
4    $u'_i \leftarrow \max(u_i - \delta, 0)$ 
5 return  $U' = \bigcup_{i=1}^n \{u'_i\}$ 

```

Algorithm 5: BinSearch function

Input : bot, top, F, n, X
Output: δ

```

1 if  $bot = top$  then
2   return  $bot$ 
3 else
4   for  $i \leftarrow 1$  to  $n$  do
5      $u'_i \leftarrow u_i - \frac{top+bot}{2}$ 
6     if  $\maxFlow < F$  then
7       return  $\text{BinSearch}(bot, \frac{top+bot}{2}, n, X, F)$ 
8     else
9       return  $\text{BinSearch}(\frac{top+bot}{2}, top, n, X, F)$ 

```

Proof of correctness - TOEXPAND. 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