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, $N^+(A) = \{B \in \mathcal{M} : DTr_{A \to B} > 0\}$. $N^+(A)$ 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, $N^-(A) = \{B \in \mathcal{M} : DTr_{B \to A} > 0\}$. $N^-(A)$ 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, $DTr_{A \to B}$).
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 $DTr_{A \to B}$ by x. This makes sense when $x \le DTr_{A \to B}$.

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

Definition 6.6 (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.

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 $DTr_{A \to N^+(A)_i}$, u_i' new $DTr_{A \to N^+(A)_i}, i \in \{1, ..., |N^+(A)|\}$.

1. The Trust Reduction on neighbour i, δ_i is defined as $\delta_i = u_i - u_i'$.

2. The Flow Reduction on neigbour 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. $||\delta_i||_1 = \sum\limits_{i \in N^+(A)} \delta_i$

2. $||\delta_i||_\infty = \max\limits_{i \in N^+(A)} \delta_i$.

Definition 6.8 (Restricted Flow)**.**
Let $A, B \in \mathcal{M}, i \in \{1, ..., |N^+(A)|\}$.

1. Let $F_{A_i \to B}$ be the flow from A to $N^+(A)_i$ as calculated by the $maxFlow(A, B)$ (x_i') when $u_i' = u_i$, $u_j' = 0 \; \forall j \in \{1, ..., |N^+(A)|\} \wedge j \neq i$.

2. Let $S \subset N^+(A)$. Let $F_{A_S \to B}$ be the sum of flows from A to S as calculated by the $maxFlow(A, B)$ $(\sum\limits_{i=1}^{|S|} x_i')$ 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, $n = |N^+(s)|, x_i, i \in \{1, ..., n\}$, 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 $\forall i \in \{1, ..., n\}, u_i' \leq x_i \Rightarrow x_i' = u_i'$.

Proof.

1. $\forall i \in \{1, ..., n\}, x_i' > u_i'$ is impossible because a flow cannot be higher than its corresponding capacity. Thus $\forall i \in \{1, ..., n\}, x_i' \leq u_i'$.

2. In the initial configuration of u_i and according to the flow problem setting, a combination of flows y_i such that $\forall i \in \{1, ..., n\}, y_i = u_i'$ is a valid, albeit not necessarily maximum, configuration with a flow $\sum\limits_{i=1}^{n} y_i$. Suppose that $\exists j \in \{1, ..., n\} : x_j' < u_j'$ as calculated by the maxFlow algorithm with the new direct trusts, u_i'. Then for the new maxFlow F' it holds that $F' = \sum\limits_{i=1}^{n} x_i' < \sum\limits_{i=1}^{n} y_i$ since $x_j' < y_j$ which is impossible because the configuration $\forall i \in \{1, ..., n\}, x_i' = y_i$ is valid since $\forall i \in \{1, ..., n\}, y_i = u_i'$ and also has a higher flow, thus the maxFlow algorithm will prefer the configuration with the higher flow. Thus we deduce that $\forall i \in \{1, ..., n\}, x_i' \geq u_i'$.

From (1) and (2) we conclude that $\forall i \in \{1, ..., n\}, x_i' = u_i'$. $\qquad\square$

Theorem 7.2 (Trust flow theorem - TOFIX)**.**
 $Tr_{A \to B} = MaxFlow_{A \to B}$ *(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 $U = \{u_1, ..., u_n\}$ with flows $X = \{x_1, ..., x_n\}$ such that $\sum_{i=1}^{n} x_i = MaxFlow_{A \to B}$ and, if severed ($\forall i \in \{1, ..., n\}\ u_i' = 0$) 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 7.3 (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.   □

Theorem 7.4 (Trust transfer theorem (flow terminology) - TOFIX)**.**
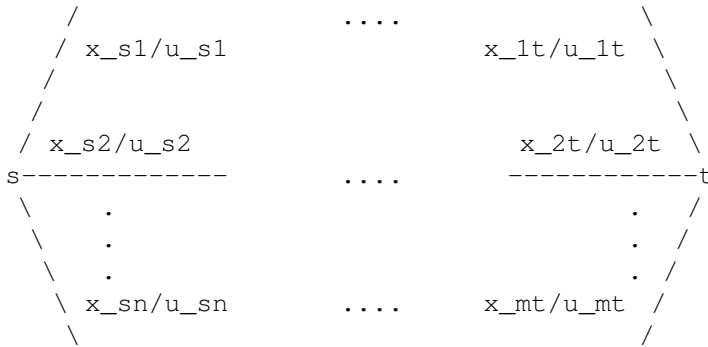*Let s source, t sink, $n = N^+(s)$*
$X_s = \{x_{s,1}, ..., x_{s,n}\}$ outgoing flows from s,
$U_s = \{u_{s,1}, ..., u_{s,n}\}$ outgoing capacities from s,
V the value to be transferred.
Nodes apart from s, t follow the honest strategy.
Obviously $maxFlow = F = \sum_{i=1}^{n} x_{s,i}$.

```
      /                   ....                \
     /  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. $\sum_{i=1}^{n} u'_{s,i} = F - V$

2. $\forall i \in \{1, ..., n\}\ u'_{s,i} \leq x_{s,i}$

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

Proof.

1. It is impossible to have $F' > F - V$ because $F' \leq \sum_{i=1}^{n} u'_{s,i} = F - V$.

2. It is impossible to have $F' < F - V$.
   Let i be a node such that $x_{s,i} > 0$ and $I = \{j \in \mathcal{M} : DTr_{i \to j} > 0\}$ the set of members that i directly

trusts with a positive value. Let $m = |I|$. In the initial graph we have $x_{s,i} = \sum_{j=1}^{m} x_{i,j}$, $F = \sum_{i=1}^{n} x_{s,i}$ and in the new graph we have $x'_{s,i} = u'_{s,i} \le x_{s,i}$, $F' = \sum_{i=1}^{n} x'_{s,i}, \forall j \, x_{i,j} \le u_{i,j} = u'_{i,j}$. We can construct a set $X'_i = \{x'_{i,j}\}$ of flows such that $x'_{i,j} \le x_{i,j}$ and $\sum_{j=1}^{m} x'_{i,j} = x'_{s,i}$. 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:

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

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

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

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

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

$\square$

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

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

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

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

Proof. We know that $x_i \le F_{A_i \to B}$, thus we can see that any increase in u'_i beyond $F_{A_i \to B}$ 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 $\forall i \in \{1, ..., n\}, u'_i \le x_i$, Trust Reduction $||\delta_i||_1$ is independent of x_i, u'_i
$\forall$ configurations of x_i

Proof. Since $\forall i \in \{1, ..., n\}, u'_i \le x_i$ 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).

$\frac{\partial x_j}{\partial x_i} = 0$ with x_i the flow from $MaxFlow \Rightarrow \forall x'_i \le x_i, \frac{\partial x_j}{\partial x_i} = 0$ *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 $F \leftarrow \sum\limits_{i=1}^{n} x_i$
2 **if** $F < V$ **then**
3 $\quad$ **return** $\perp$
4 $Fcur \leftarrow F$
5 **for** $i \leftarrow 1$ *to* n **do**
6 $\quad$ $u'_i \leftarrow x_i$
7 $i \leftarrow 1$
8 **while** $Fcur > F - V$ **do**
9 $\quad$ $reduce \leftarrow min(u'_i, Fcur - V)$
10 $\quad$ $Fcur \leftarrow Fcur - reduce$
11 $\quad$ $u'_i \leftarrow u'_i - reduce$
12 $\quad$ $i \leftarrow i + 1$
13 **return** $U' = \bigcup\limits_{i=1}^{n} \{u'_i\}$

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

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

1 $F \leftarrow \sum\limits_{i=1}^{n} x_i$
2 **if** $F < V$ **then**
3 $\quad$ **return** $\perp$
4 **for** $i \leftarrow 1$ *to* n **do**
5 $\quad$ $u'_i \leftarrow x_i$
6 $reduce \leftarrow \frac{V}{n}$
7 $reduction \leftarrow 0$
8 $empty \leftarrow 0$
9 $i \leftarrow 0$
10 **while** $reduction < V$ **do**
11 $\quad$ **if** $u'_i > 0 \wedge x_i < reduce$ **then**
12 $\quad\quad$ $empty \leftarrow empty + 1$
13 $\quad\quad$ $reduce = reduce + \frac{x_i - reduce - u'_i}{n - empty}$
14 $\quad\quad$ $reduction \leftarrow reduction + u'_i$
15 $\quad\quad$ $u'_i \leftarrow 0$
16 $\quad$ **else if** $x_i \geq reduce$ **then**
17 $\quad\quad$ $reduction \leftarrow reduction + u'_i - (x_i - reduce)$
18 $\quad\quad$ $u'_i \leftarrow x_i - reduce$
19 $\quad$ $i \leftarrow (i+1) \, mod \, n$
20 **return** $U' = \bigcup\limits_{i=1}^{n} \{u'_i\}$

Algorithm 3: Proportional equality trust transfer

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

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

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

Input : x_i flows, $n = |N^+(s)|$, V value

Output: u_i' capacities

1 **return** $U' = \bigcup\limits_{i=1}^{n} \{u_i'\}$

Proof of correctness - TOEXPAND. 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 $\sum\limits_{i=1}^{n} u_i' = F - V$. $\qquad\square$

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

8 Further Research

9 References