# 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-of-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** (Saturation theorem).
*Let $s$ source, $x_i, i \in \{1, ..., |N(s)|\}$, flows to $s$'s neighbours as calculated by the maxFlow algorithm, $u'_i$ new direct trusts to the $|N(s)|$ 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 $\forall i \in \{1, ..., |N(s)|\}, u'_i \leq x_i \Rightarrow x'_i = u'_i$.*

*Proof.*

1. $\forall i \in \{1, ..., |N(s)|\}, x'_i > u'_i$ is impossible because a flow cannot be higher than its corresponding capacity. Thus $\forall i \in \{1, ..., |N(s)|\}, x'_i \leq u'_i$.

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

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

**Theorem 7.2** (Trust flow theorem).
$Tr_{A \to B} = MaxFlow_{A \to B}$ *(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)).
*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 $maxFlow = F = \sum_{i=1}^{n} x_{s,i}$.*

---

```
         /              ....          \
        / 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. $\sum_i u'_{s,i} = F - V$

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 $F' \leq \sum u'_{s,i} = F - V$.

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 $x'_{s,i} = u'_{s,i} \leq x_{s,i}, F' = \sum_i x'_{s,i}, \forall j x_{i,j} \leq u_{i,j} = u'_{i,j}$. 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:
$\forall j \in \{1, ..., k\}, x'_{i,j} = x_{i,j}$ with $k$ such that

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

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

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

□

**Corollary 7.1** (Requirement for $\sum_i u'_{s,i} = F - V$, $u'_{s,i} \leq x_{s,i}$)**.**
*In the setting of 7.4, it is impossible to have* $maxFlow' = F - V$ *if* $\sum_i u'_{s,i} > F - V \wedge \forall i \in \{1, ..., n\}, u'_{s,i} \leq x_{s,i}$.

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

**Theorem 7.5** (Trust-saving Theorem)**.**
$\forall i \in \{1, ..., n\}, u'_i = F_{A_i \to B} \Leftrightarrow u'_i = u_i$

*Proof.* We know that $x_i \leq F_{A_i \to B}$, 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. □

**Theorem 7.6** (Invariable trust reduction with naive algorithms)**.**
*If* $\forall i \in \{1, ..., n\}, u'_i \leq x_i$, *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 $\forall i \in \{1, ..., n\}, u'_i \leq x_i$ 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 $x'_i, u'_i$ □

**Theorem 7.7** (Dependence impossibility theorem)**.**
$\frac{\partial x_j}{\partial x_i} = 0$ *with* $x_i$ *the flow from MaxFlow* $\Rightarrow \forall x'_i \leq x_i, \frac{\partial x_j}{\partial x_i} = 0$ *ceteris paribus*

*Proof.* TODO □

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

$F \leftarrow \sum_{i=1}^{n} x_i$
**if** $F < V$ **then**
  **return** $\perp$
$Fcur \leftarrow F$
**for** $i \leftarrow 1$ *to* $n$ **do**
  $u'_i \leftarrow x_i$
$i \leftarrow 1$
**while** $Fcur > F - V$ **do**
  $reduce \leftarrow min(u'_i, Fcur - V)$
  $Fcur \leftarrow Fcur - reduce$
  $u'_i \leftarrow u'_i - reduce$
  $i \leftarrow i + 1$
**return** $U' = \bigcup_{i=1}^{n} \{u'_i\}$

**Algorithm 2:** Absolute equality trust transfer

**Input :** $x_i$ flows, $n$ flows number, $V$ value
**Output:** $u'_i$ capacities

$F \leftarrow \sum_{i=1}^{n} x_i$
**if** $F < V$ **then**
  **return** $\perp$
**for** $i \leftarrow 1$ *to* $n$ **do**
  $u'_i \leftarrow x_i$
$reduce \leftarrow \frac{V}{n}$
$reduction \leftarrow 0$
$empty \leftarrow 0$
$i \leftarrow 0$
**while** $reduction < V$ **do**
  **if** $u'_i > 0 \wedge x_i < reduce$ **then**
    $empty \leftarrow empty + 1$
    $reduce = reduce + \frac{x_i - reduce - u'_i}{n - empty}$
    $reduction \leftarrow reduction + u'_i$
    $u'_i \leftarrow 0$
  **else if** $x_i \geq reduce$ **then**
    $reduction \leftarrow reduction + u'_i - (x_i - reduce)$
    $u'_i \leftarrow x_i - reduce$
  $i \leftarrow (i + 1) mod\, n$
**return** $U' = \bigcup_{i=1}^{n} \{u'_i\}$

**Algorithm 3:** Proportional equality trust transfer

**Input :** $x_i$ flows, $n$ flows number, $V$ value
**Output:** $u'_i$ capacities

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

*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