Trust Is Risk:
A Decentralized Financial Trust Platform

Orfeas Stefanos Thyfronitis Litos[1] and Dionysis Zindros[2,⋆]

[1] National Technical University of Athens
orfeas.litos@hotmail.com
[2] University of Athens
dionyziz@di.uoa.gr

Abstract. Reputation in centralized systems uses stars and review-based trust. Such systems require manual intervention and secrecy to avoid manipulation. In autonomous and open source decentralized systems this luxury is not available. Previous peer-to-peer reputation systems 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 associates. Indirect trust is subsequently defined transitively. This enables formal game theoretic arguments pertaining to risk analysis. We prove that risk and max flows are equivalent in our model. Our system allows for concrete financial decisions on the monetary amount a pseudonymous party can be trusted with. Through algorithmic trust redistribution, the risk incurred from making a purchase from a pseudonymous party in this manner remains invariant.

1 Introduction

Modern marketplaces can be roughly categorized as centralized and decentralized. Two major examples are ebay and OpenBazaar. ebay is centralized and as such it is vulnerable to ddos attacks [1] and can be considered as a single point of failure, it charges fees for the use of its services [2] and maintains a private database of personal data, but it can give money-back guarantees [3] since it is run by a single company that has a financial advantage in keeping its clients satisfied. On the other hand, OpenBazaar is a decentralized platform built on bitcoin [6], where individual stores or its search engine are vulnerable to ddos attacks [1], but not the platform as a whole. Additionally, it does not charge fees for its usage [4] and there is no central agent recording all the transactions alongside with private data [4] but it is possible for a buyer or a seller colluding with a moderator to scam the third party and there exists no central authority able

⋆ Research supported by ERC project CODAMODA, project #259152

to verify the truth of her claim and reimburse her [5]. Even though trust (or distrust) should be directed to each store individually, it is very likely that the whole platform will be discarded as untrusted by the scammed party.

Our goal is to create a distributed marketplace where the trust each user gives to the rest of the users is quantifiable, measurable and expressable in monetary terms. The central concept used throughout this paper is trust-as-risk, or the proposition that a user's A *trust* to another user B is defined to be the *maximum sum of money* A can lose when B is free to choose any strategy she wants. To flesh out trust-as-risk, we will use *lines of credit* as proposed by Washington Sanchez [12]. Joining the network will be done by explicitly entrusting a certain amount of money to another user. If the second user has already entrusted an amount of money to a store, then we trust indirectly the store, thus we can engage in economic interaction with said store.

We thus propose a new kind of wallet where coins are not stored locally, but are placed in 1-of-2 multisigs, a bitcoin construction that permits any one of two pre-designated users to spend the coins contained therein [11]. We will use the notation $1/\{A, B\}$ to represent a 1-of-2 multisig that can be spent by either of players A, B.

Our approach drastically changes the user experience. A user no more has to worry about stars and ratings to develop trust towards a store, she can simply check the trust *flowing* from her to the store (which will be a number expressed in bitcoins) and if this number exceeds the price of the product, she is safe to complete the transaction after modifying her direct trust towards her friends in an appropriate way, a process that can be handled by one of several algorithms that we propose, or in any custom way the user chooses to. On the other side, there is no guarantee that the store will complete is part of the transaction and no central authority can reimburse the money. However, it is possible for the defrauded user to make up for her loss by in turn stealing from other users that trust her, an action that will tautologically reduce their trust to her. The fact that this system can function in a completely distributed fashion will become clear in the following sections, along with the positive result that it is in principle Sybil resilient.

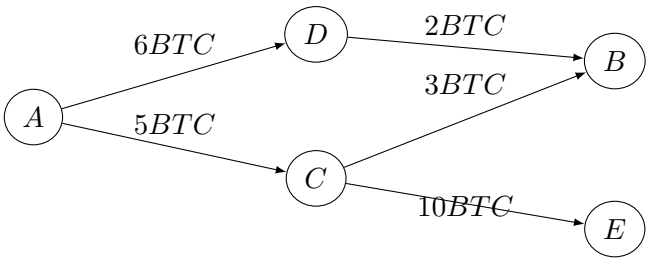
There are several incentives for a user to join this network. First of all, they can have access to a store that is otherwise unaccessible. Moreover, two friends can formalize their mutual trust by entrusting the same amount to each other. A large company that casually subcontracts other companies to complete various tasks can express its trust towards them

using this method. A government can choose to entrust its citizens with money and confront them using its legal arsenal if they make irresponsible use of this trust. A bank can provide loans as outgoing and manage savings as incoming trust and thus has a unique opportunity of expressing in a formal and absolute way its credence by publishing its incoming trust. Last but not least, the network can be viewed as a possible field for investment and speculation since it constitutes a completely new area for financial activity.

2 A detailed example

The state of the game at any given moment can be represented by a directed weighted graph where each node represents a pseudonymous identity that is considered to correspond to exactly one user and each edge represents a direct trust, the weight of which is the value which the head entrusts the tail with.

TrustIsRisk game graph



Each node has a corresponding non-negative number that represents its capital. A node's capital is the total value that the node directly and exclusively controls. More technically, it is the number of bitcoins contained in P2PKH transactions in the UTXO [7] of which the private keys are controlled by the node. A rational player would like to maximize her capital in the long term. Let's suppose that player A wants to buy a product that costs 10 from player B, but A has not given any direct trust to B.

Does A's friends trust B enough so that A can buy the product? In other words, should B decide to steal all her incoming direct trust and stop trusting directly other users and the rest of the players choose the conservative strategy of stealing from their respective incoming trusts just enough to replenish their loss [rationale needed], what would the eventual loss of A be? If it can be higher than the value of the product, then A is already exposed to a higher risk to B, thus paying for the product can be done without A being exposed to any additional risk towards B, as long as A reduces the direct trust to her friends enough to reduce the risk by the value of the product. If the transaction completes in a satisfying manner, A can replenish the reduced trust to her friends, or can even choose to start directly trusting B.

Suppose that at the moment A is considering of performing a payment to him, B really decides to depart from the network, taking with him all the coins he can. A chain reaction starts throughout the network because each player that B exploited is expected to steal enough value from other users that directly trust her to replenish all the value she lost to B, the newly exploited users will act in the same way and so on, until a new equilibrium is reached (1). Going into detail, [Image]
(Completion of example)

3 Definitions

Definition 1 (Graph).
TrustIsRisk is represented by a sequence of weighted directed graphs $(\mathcal{G}_j)$ *where* $\mathcal{G}_j = (\mathcal{V}_j, \mathcal{E}_j), j \in \mathbb{N}$. *Also, since the graphs are weighted, there exists a sequence of functions* (c_j) *with* $c_j : \mathcal{E}_j \to \mathbb{R}^+$.

Definition 2 (Players).
The set $\mathcal{V}_j = V(\mathcal{G}_j)$ *is the set of all players in the network, otherwise understood as the set of all pseudonymous identities.*

Definition 3 (Capital of A, Cap_A).
The capital of A, Cap_A *is defined as the total amount of value that exists in P2PKH in the UTXO and can be spent by A. We also define* $Cap_{A,j}$ *as the total amount of value that exists in P2PKH in the UTXO and can be spent by A during turn* j.

Definition 4 (Direct Trust from A to B After Turn j, $DTr_{A\to B,j}$).
Direct trust from A to B at the end of turn j, $DTr_{A\to B,j}$, *is defined as*

the total amount of value that exists in 1/$\{A, B\}$ multisigs in the UTXO in the end of turn j, where the money is deposited by A.

$$DTr_{A\to B,j} = \begin{cases} c_j(A,B), & if\ (A,B) \in \mathcal{E}_j \\ 0, & if\ (A,B) \notin \mathcal{E}_j \end{cases} \tag{1}$$

An algorithm that has access to the graph $\mathcal{G}_j$ has implicitly access to all direct trusts of this graph. The exception are the oracles, which in this case have access only to their incoming and outgoing direct trusts.

Definition 5 ((In/Out) Neighbourhood of A in Turn j).

1. *Let $N^+(A)_j$ be the set of players B that A directly trusts with any positive value at the end of turn j. More formally,*

$$N^+(A)_j = \{B \in \mathcal{V}_j : DTr_{A\to B,j} > 0\} \ . \tag{2}$$

   $N^+(A)_j$ is called out neighbourhood of A on turn j. Let $S \subset \mathcal{V}_j$.

$$N^+(S)_j = \bigcup_{A\in S} N^+(A)_j \tag{3}$$

2. *Let $N^-(A)_j$ be the set of players B that directly trust A with any positive value at the end of turn j. More formally,*

$$N^-(A)_j = \{B \in \mathcal{V}_j : DTr_{B\to A,j} > 0\} \ . \tag{4}$$

   $N^-(A)_j$ is called in neighbourhood of A on turn j. Let $S \subset \mathcal{V}_j$.

$$N^-(S)_j = \bigcup_{A\in S} N^-(A)_j \tag{5}$$

3. *Let $N(A)_j$ be the set of players B that either directly trust or are directly trusted by A with any positive value at the end of turn j. More formally,*

$$N(A)_j = N^+(A)_j \cup N^-(A)_j \ . \tag{6}$$

   $N(A)_j$ is called neighbourhood of A on turn j. Let $S \subset \mathcal{V}_j$.

$$N(S)_j = N^+(S)_j \cup N^-(S)_j \tag{7}$$

4. *Let $S \subset \mathcal{V}_j$. Let $N(A)_{j,i}$ (respectively $N^+(A)_{j,i}, N^-(A)_{j,i}, N(S)_{j,i}$, $N^+(S)_{j,i}, N^-(S)_{j,i}$) be the i-th element of set $N(A)_j$ (respectively of $N^+(A)_j, N^-(A)_j, N(S)_j, N^+(S)_j, N^-(S)_j$), according to an arbitrary but fixed enumeration of the set players.*

Definition 6 (Total Incoming/Outgoing Trust of A in Turn j).

$$in_{A,j} = \sum_{v \in N^-(A)_j} DTr_{v \to A,j} \tag{8}$$

$$out_{A,j} = \sum_{v \in N^+(A)_j} DTr_{A \to v,j} \tag{9}$$

Definition 7 (Turns).
The game we are describing is turn-based. In each turn j exactly one player $A \in \mathcal{V}, A = Player(j)$, chooses an action (according to a certain strategy) that can be one of the following, or a finite combination thereof:

1. *Steal value $y_B, 0 \leq y_B \leq DTr_{B \to A,j-1}$ from $B \in N^-(A)$.*

$$DTr_{B \to A,j} = DTr_{B \to A,j-1} - y_B \ \ (Steal(y_B, B)) \tag{10}$$

2. *Add value $y_B, -DTr_{A \to B,j-1} \leq y_B$ to $B \in \mathcal{V}$.*

$$DTr_{A \to B,j} = DTr_{A \to B,j-1} + y_B \ \ (Add(y_B, B)) \tag{11}$$

When $y_B < 0$, we say that A reduces her trust to B by $-y_B$, when $y_B > 0$, we say that A increases her trust to B by y_B. If $DTr_{A \to B,j-1} = 0$, then we say that A starts directly trusting B.

If player A chooses no action in her turn, we say that she passes her turn. Also, let Y_{st}, Y_{add} be the total value to be stolen and added respectively by A in her turn, j. For a turn to be feasible, it must hold that

$$Y_{add} - Y_{st} \leq Cap_{A,j-1} \ . \tag{12}$$

Capital is updated in every round:

$$Cap_{A,j} = Cap_{A,j-1} + Y_{st} - Y_{add} \ . \tag{13}$$

Moreover, player A is not allowed to choose two actions of the same kind against the same player in the same turn.
The set of actions a player makes in turn j is represented with $Turn_j$. The new graph that emerges by applying the actions on $\mathcal{G}_{j-1}$ is $\mathcal{G}_j$.

Definition 8 (Previous/Next Turn of a Player).
Let $j \in \mathbb{N}$ a turn with $Player(j) = A$. We define $prev(j), next(j)$ as the previous and next turn that A is chosen to play respectively. If j is the first turn that A plays, $prev(j) = 0$. More formally, if

$$P = \{k \in \mathbb{N} : k < j \wedge Player(k) = A\} \text{ and} \tag{14}$$

$$N = \{k \in \mathbb{N} : k > j \wedge Player(k) = A\} \ , \tag{15}$$

then we define $prev(j), next(j)$ *as follows:*

$$prev(j) = \begin{cases} \max P, & P \neq \emptyset \\ 0, & P = \emptyset \end{cases} \tag{16}$$

$$next(j) = \min N \tag{17}$$

$next(j)$ *is always well defined with the assumption that eventually everybody plays.*

Definition 9 (A is Stolen x).
Let j, j' *be two consecutive turns of* A*. We say that* A *has been stolen a value* x *between* j *and* j' *if*

$$out_{A,j} - out_{A,j'} > 0 \ . \tag{18}$$

If turns are not specified, we implicitly refer to the current and the previous turns.

Definition 10 (History).
We define History, $\mathcal{H} = (\mathcal{H}_j)$*, as the sequence of all the tuples containing the sets of actions and the corresponding player.*

$$\mathcal{H}_j = (Player(j), Turn_j) \tag{19}$$

Definition 11 (Idle Strategy).
A player A *is said to follow the idle strategy if she passes in her turn. More formally, it holds that*

$$\left.\begin{array}{l} Strategy(A) = Idle \\ j \in \mathbb{N} : Player(j) = A \end{array}\right\} \Rightarrow Turn_j = \emptyset \ . \tag{20}$$

```
Idle Oracle
Input : previous graph G_{j-1}
Output : Turn_j
idleOracle(G_{j-1}) :
1   return(∅)
```

Definition 12 (Evil Strategy).
A player A *is said to follow the evil strategy if she steals value*

$$y_v = DTr_{v \to A, j-1} \, \forall \, v \in N^-(A)_j \tag{21}$$

(steals all incoming direct trust) and reduces her trust to w by

$$DTr_{A\to w,j-1}\ \forall\, w \in N^{+}(A)_j \tag{22}$$

(nullifies her outgoing direct trust) in her turn. More formally, it holds that

$$\left.\begin{array}{l} Strategy(A) = Evil \\ j \in \mathbb{N} : Player(j) = A \end{array}\right\} \Rightarrow Turn_j = Steals \cup Adds \ \ , \ where \tag{23}$$

$$\begin{aligned} Steals &= \bigcup_{v \in N^{-}(A)_{j-1}} \{Steal(DTr_{v\to A,j-1}, v)\} \ and \\ Adds &= \bigcup_{w \in N^{+}(A)_{j-1}} \{Add\,(-DTr_{A\to w,j-1}, w)\} \ . \end{aligned} \tag{24}$$

We again note that $N(A)_{j-1} = N(A)_j$.

```
Evil Oracle
Input : previous graph G_{j-1}
Output : Turn_j
evilOracle(G_{j-1}) :
1  Turn_j = ∅
2  for (v ∈ N^-(A)_{j-1})
3    Turn_j = Turn_j ∪ {Steal(DTr_{v→A,j-1}, v)}
4  for (w ∈ N^+(A)_{j-1})
5    Turn_j = Turn_j ∪ {Add(−DTr_{A→v,j-1}, w)}
6  return(Turn_j)
```

Definition 13 (Conservative Strategy).
Let j be the current turn and x the value that has been stolen from player A since the previous turn she played. Player A is said to follow the conservative strategy if she replenishes her lost value by stealing from others that trust her as much as she can up to x and she takes no other action. More formally, let $j' = prev(j)$, $Damage_j = out_{A,j'} - out_{A,j-1}$ and $N^{-}(A)_j = \{v_1, \ldots, v_k\}$. It holds that

$$\left.\begin{array}{l} Strategy(A) = Conservative \\ j \in \mathbb{N} : Player(j) = A \end{array}\right\} \Rightarrow$$
$$\Rightarrow Turn_j = \begin{cases} \emptyset, & Damage_j \le 0 \\ \bigcup\limits_{i=1}^{k} \{Steal\,(y_i, v_i)\}, & Damage_j > 0 \end{cases} . \tag{25}$$

In the second case, it holds that

$$\sum_{i=1}^{k} y_i = \min\left(in_{A,j-1}, Damage_j\right) \quad . \tag{26}$$

The oracle remembers `PrevOutTrust` $= out_{A,j'}$ for $j' = prev\,(j)$ and can observe all incoming and outgoing direct trusts of player A. We note that $N\,(A)_{j-1} = N\,(A)_j$.

```
Conservative Oracle
Input : previous graph G_{j-1}
Output : Turn_j
consOracle(G_{j-1}) :
1   NewOutTrust = out_{A,j-1}
2   NewInTrust = in_{A,j-1}
3   Damage = PrevOutTrust(A) - NewOutTrust
4                                   #PrevOutTrust(A) = out_{A,prev(j)}
5   PrevOutTrust(A) = NewOutTrust
6   if (Damage > 0)
7     if (Damage >= NewInTrust)
8       Turn_j = ∅
9       for (v ∈ N^-(A)_{j-1})
10        Turn_j = Turn_j ∪ {Steal(DTr_{v→A,j-1}, v)}
11    else
12      y = SelectSteal(G_j, A, Damage) #y = (y_1,...,y_{|N^-(A)_{j-1}|})
13      Turn_j = ∅
14      for (i ∈ [|N^-(A)_{j-1}|])
15        Turn_j = Turn_j ∪ {Steal(y_i, N^-(A)_{j-1,i})}
16  else
17    Turn_j = ∅
18  return(Turn_j)
```

`SelectSteal()` returns y_i with $i \in [|N^-\,(A)_j\,|]$ such that

$$\sum_{i=1}^{|N^-(A)_j|} y_i = Damage \wedge \forall i, y_i \leq DTr_{N^-(A)_{j,i} \to A} \quad . \tag{27}$$

Each conservative player can arbitrarily define how `SelectSteal()` distributes the *Steal* () actions each time she calls the function, as long as the above restriction is respected. As we can see, the definition covers a multitude of options for the conservative player, since in case $0 < Damage_j <$

$in_{A,j-1}$ she can choose to distribute the $Steal\,()$ actions in any way she chooses.

The rationale behind this strategy arises from a real-world common situation. Suppose there is a client, an intermediary and a producer. The client entrusts some value to the intermediary so that the latter can buy the desired product from the producer and deliver it to the client. The intermediary in turn entrusts an equal value to the producer, who needs the value upfront to be able to complete the production process. However the producer eventually does not give the product neither reimburses the value, due to bankruptcy or decision to exit the market with an unfair benefit. The intermediary can choose either to reimburse the client and suffer the loss, or refuse to return the money and lose the client's trust. The latter choice for the intermediary is exactly the conservative strategy. It is used throughout this work as a strategy for all the intermediary players because it models effectively the worst-case scenario that a client can face after an evil player decides to steal everything she can and the rest of the players do not engage in evil activity.

Definition 14 (Indirect Trust, $Tr_{A\to B,j}$).
The indirect trust from A to B after turn j is defined as the maximum possible value that can be stolen from A if B follows the evil strategy, A follows the idle strategy and everyone else ($\mathcal{V}\setminus\{A,B\}$) follows the conservative strategy. More formally, if

$$\begin{gathered} Strategy\,(A) = Idle \wedge Strategy\,(B) = Evil \wedge \\ \wedge \forall v \in \mathcal{V}\setminus\{A,B\}, Strategy\,(v) = Conservative \end{gathered} \tag{28}$$

and choices are the different actions between which the conservative players can choose, then

$$Tr_{A\to B,j} = \max_{j':j'>j,choices} \left[out_{A,j} - out_{A,j'}\right] \tag{29}$$

Definition 15 (Indirect Trust to multiple players, $Tr_{A\to S,j}$).
The indirect trust from player A to a set of players, $S \subset \mathcal{V}$ is defined as the maximum possible value that can be stolen from A if all players in S follow the evil strategy, A follows the idle strategy and everyone else ($\mathcal{V}\setminus(S\cup\{A\})$) follows the conservative strategy. More formally, if $S \subset \mathcal{V}$,

$$\begin{gathered} Strategy\,(A) = Idle \wedge \forall E \in S, Strategy\,(E) = Evil \wedge \\ \wedge \forall v \in \mathcal{V}\setminus(S\cup\{A\}), Strategy\,(v) = Conservative \end{gathered} \tag{30}$$

and choices are the different actions between which the conservative players can choose, then

$$Tr_{A\to S,j} = \max_{j':j'>j,configurations} \left[out_{A,j} - out_{A,j'}\right] \tag{31}$$

4 Theorems-Algorithms

The following algorithm has read access to direct trusts in $\mathcal{G}_{j-1}$ and write access to direct trusts in $\mathcal{G}_j$.

```
Execute Turn
Input : old graph G_{j-1}, player A ∈ V_{j-1}, old capital
   Cap_{A,j-1}, ProvisionalTurn
Output : new graph G_j, new capital Cap_{A,j}, new history H_j
executeTurn(G_{j-1}, A, Cap_{A,j-1}, ProvisionalTurn) :
  (Turn_j, NewCap) = validateTurn(G_{j-1}, A, Cap_{A,j-1},
     ProvisionalTurn)
  return(commitTurn(G_{j-1}, A, Turn_j, NewCap))

Validate Turn
Input : old graph G_{j-1}, player A ∈ V_{j-1}, old capital
   Cap_{A,j-1}, ProvisionalTurn
Output : Turn_j, new capital Cap_{A,j}
validateTurn(G_{j-1}, A, Cap_{A,j-1}, ProvisionalTurn) :
  Y_st = 0
  Y_add = 0
  for (action ∈ ProvisionalTurn)
    action match do
      case Steal(y, w) do
        if (y > DTr_{w→A,j-1} || y < 0)
          return(∅, Cap_{A,j-1})
        else
          Y_st = Y_st + y
      case Add(y, w) do
        if (y < -DTr_{A→w,j-1})
          return(∅, Cap_{A,j-1})
        else
          Y_add = Y_add + y
  if (Y_add - Y_st > Cap_{A,j-1})
    return(∅, Cap_{A,j-1})
  else
    return(ProvisionalTurn, Cap_{A,j-1} + Y_st - Y_add)
```

```
Commit Turn
Input : player old graph G_{j-1}, A ∈ V_{j-1}, Turn_j, NewCap
Output : new graph G_j, new capital Cap_{A,j}, new history H_j
commitTurn(G_{j-1}, A, Turn_j, NewCap) :
  for ((v, w) ∈ E_j)
    DTr_{v→w,j} = DTr_{v→w,j-1}
  for (action ∈ Turn_j)
    action match do
      case Steal(y,, w) do
        DTr_{w→A,j} = DTr_{w→A,j-1} − y
      case Add(y, w) do
        DTr_{A→w,j} = DTr_{A→w,j-1} + y
  Cap_{A,j} = NewCap
  H_j = (A, Turn_j)
  return(G_j, Cap_{A,j}, H_j)
```

```
TrustIsRisk Game
j = 0
while (True)
  j = j + 1
  v ←$ V_j
  ProvisionalTurn = vOracle(G_{j-1})
  (G_j, Cap_{v,j}, H_j) = executeTurn(G_{j-1}, v, Cap_{v,j-1},
      ProvisionalTurn)
```

We continue with a very useful possible evolution of the game, the Transitive Game. In turn 0, there is already a network in place. Also, all players apart from A and E follow the conservative strategy. Furthermore, the set of players is not modified throughout the Transitive Game, thus we can refer to $\mathcal{V}_j$ for any turn j as $\mathcal{V}$. These conventions will hold whenever we use the Transitive Game.

```
Transitive Game
Input : graph G_0, A ∈ V idle player, E ∈ V evil player
Output : history H
Angry = ∅
Happy = V \ {A, E}
Sad = ∅
for (v ∈ V \ {E})
  Loss_v = 0
j = 0
```

```
while (True)
  j = j + 1
  v ←$ V \ {A}
  Turn_j = vOracle(G_{j-1})
  executeTurn(G_{j-1}, Cap_{v,j-1}, Turn_j)
  for (action ∈ Turn_j)
    action match do
      case Steal(y, w) do
        exchange = y
        Loss_w = Loss_w + exchange
        if (v != E)
          Loss_v = Loss_v - exchange
        if (w != A)
          Happy = Happy \ {w}
          if (in_{w,j} == 0)
            Sad = Sad ∪ {w}
          else
            Angry = Angry ∪ {w}
  if (v != E)
    Angry = Angry \ {v}
    if (Loss_v > 0)
      Sad = Sad ∪ {v}                    #in_{v,j} should be zero
    if (Loss_v == 0)
      Happy = Happy ∪ {v}
```

Let j_0 be the first turn on which E is chosen to play. Until then, all players will pass their turn since nothing has been stolen yet (see Appendix (5) for a formal proof of this simple fact). Moreover, let $j' = prev\,(j)$ and $\{v_1, \ldots, v_k\} = N^-\,(A)_j$. Given that

$$Damage_{v,j} = out_{v,j'} - out_{v,j} \ , \tag{32}$$

the algorithm generates turns:

$$Turn_j = \begin{cases} \emptyset, & Damage_{v,j-1} = 0 \\ \bigcup\limits_{i=1}^{k} \{Steal\,(y_i, v_i)\}, & Damage_{v,j-1} > 0 \end{cases} \ . \tag{33}$$

In the second case, it is

$$\sum_{i=1}^{k} y_i = \min\,(in_{v,j-1}, Damage_{v,j-1}) \ . \tag{34}$$

From the definition of $Damage_{v,j}$ and knowing that no strategy in this case can increase any direct trust, it is obvious that $Damage_{v,j} \geq 0$. Also, we can see that $Loss_{v,j} \geq 0$ because if $Loss_{v,j} < 0$, then v has stolen more value than she has been stolen, thus she would not be following the conservative strategy.

Theorem 1 (Trust Convergence Theorem).
Consider a Transitive Game and let j_0 be the first turn that E is chosen to play. Given that all players eventually play, there exists a turn $j' > j_0$ such that

$$\forall j \geq j', Turn_j = \emptyset \ . \tag{35}$$

Proof Sketch. If the game didn't converge, the *Steal* () actions would continue forever without reduction of the amount stolen over time, thus they would reach infinity. However this is impossible, since there exists only finite total trust. For the complete proof, see Appendix (6.3). □

In games where there exists one evil E, one idle player A and the rest of the players are conservative, we define $Loss_A = Loss_{A,j}$, where j is a turn that the game has converged. It is important to note that $Loss_A$ is not the same for repeated executions of this kind of game, since the order in which players are chosen may differ between executions and the conservative players are free to choose which incoming trusts they will steal and how much from each.

Let G be a weighted directed graph. According to [9] p. 709, if we consider each edge's capacity as its weight ($\forall e \in E(G), c_e = c(e)$), we say that a flow assignment $X = [x_{vw}]_{V(FG)\times V(FG)}$ with a source A and a sink B is valid when:

$$\forall (v, w) \in E(FG), x_{vw} \leq c_{vw} \tag{36}$$

and

$$\forall v \in V(FG) \setminus \{A, B\}, \sum_{w \in N^+(v)} x_{wv} = \sum_{w \in N^-(v)} x_{vw} \ . \tag{37}$$

Lemma 1 (MaxFlows Are Transitive Games).
Let $\mathcal{G}_{j_0}$ be a game graph at a specific turn j_0, let $A, E \in \mathcal{V}_{j_0}$ and $maxFlow\,(A, E)$ the maximum flow from A to E executed on $\mathcal{G}_{j_0}$. There exists an execution of `TransitiveGame`$(\mathcal{G}_{j_0}, A, E)$ such that

$$maxFlow\,(A, E) \leq Loss_A \ . \tag{38}$$

Proof Sketch. The execution will contain all flows from the $MaxFlow\,(A, E)$ as equivalent $Steal\,()$ actions. The players will play in turns, moving from E back to A. Each player will steal from his predecessors as much as was stolen from her. The flows and the conservative strategy share the property that the total input is equal to the total output. For the complete proof, see Appendix (6.3). □

Lemma 2 (Transitive Games Are Flows).
Let $\mathcal{H} =$`TransitiveGame`$(\mathcal{G}, A, E)$ *for some game graph* $\mathcal{G}$ *and* $A, E \in \mathcal{V}$. *There exists a valid flow* $X = \{x_{wv}\}_{\mathcal{V}\times\mathcal{V}}$ *on* $\mathcal{G}_0$ *such that*

$$\sum_{v\in\mathcal{V}} x_{Av} = Loss_A \quad . \tag{39}$$

Proof Sketch. If we remove the sad players from the game, the $Steal\,()$ actions that remain constitute a valid flow from A to E. For the complete proof, see Appendix (6.3). □

Theorem 2 (Trust Flow Theorem).
Let $\mathcal{G}$ *be a game graph and* $A, B \in \mathcal{V}$. *It holds that*

$$Tr_{A\to B} = maxFlow\,(A, B) \quad . \tag{40}$$

Proof.
From lemma 1 we see that there exists an execution of the Transitive Game such that

$$Loss_A = maxFlow\,(A, B) \quad . \tag{41}$$

Since $Tr_{A\to B}$ is the maximum loss that A can suffer after the convergence of the Transitive Game, we see that

$$Tr_{A\to B} \geq maxFlow\,(A, B) \quad . \tag{42}$$

Moreover, there exists an execution of the Transitive Game such that

$$Tr_{A\to B} = Loss_A \quad . \tag{43}$$

From lemma 2, this execution corresponds to a flow. Thus

$$Tr_{A\to B} \leq maxFlow\,(A, B) \quad . \tag{44}$$

The theorem follows from (42) and (44). □

Note: The maxFlow is the same in the following two cases: When a player chooses the evil strategy and when the same player chooses a variation of the evil strategy where she does not nullify her outgoing direct trust.

Theorem 3 (Trust to Multiple Players).
Let $S \subset \mathcal{V}$ and T auxiliary player such that

$$\forall B \in S, DTr_{B \to T} = \infty \ . \tag{45}$$

It holds that

$$\forall A \in \mathcal{V} \setminus S, Tr_{A \to S} = maxFlow\,(A, T) \ . \tag{46}$$

Proof. If T chooses the evil strategy and all players in S play according to the conservative strategy, they will have to steal all their incoming direct trust, thus they will act in a way identical to following the evil strategy as far as $MaxFlow$ is concerned. The theorem follows thus from the Trust Flow Theorem. □

One of the primary aims of this system is to mitigate the danger for sybil attacks whilst maintaining fully decentralized autonomy. Let Eve be a possible attacker.

Definition 16 (Corrupted Set).
Let $\mathcal{G}$ be a game graph and let Eve have a set of players $\mathcal{B} \subset \mathcal{V}$ corrupted, so that she fully controls their outgoing direct trusts to any player in $\mathcal{V}$ and can also steal all incoming direct trust to players in $\mathcal{B}$. We call this the corrupted set. The players $\mathcal{B}$ are considered to be legitimate before the corruption, thus they may be directly trusted by any player in $\mathcal{V}$.

Definition 17 (Sybil Set).
Let $\mathcal{G}$ be a game graph. Since participation in the network does not require any kind of registration, Eve can create any number of players. We will call the set of these players $\mathcal{C}$, or Sybil set. Moreover, Eve can arbitrarily set the direct trusts of any player in $\mathcal{C}$ to any player and can also steal all incoming direct trust to players in $\mathcal{C}$. However, players $\mathcal{C}$ can be directly trusted only by players $\mathcal{B} \cup \mathcal{C}$ but not by players $\mathcal{V} \setminus (\mathcal{B} \cup \mathcal{C})$, where $\mathcal{B}$ is a set of players corrupted by Eve.

Definition 18 (Collusion).
Let $\mathcal{G}$ be a game graph. Let $\mathcal{B} \subset \mathcal{V}$ be a corrupted set and $\mathcal{C} \subset \mathcal{V}$ be a Sybil set, both controlled by Eve. The tuple $(\mathcal{B}, \mathcal{C})$ is called a collusion

and is entirely controlled by a single entity in the physical world. From a game theoretic point of view, players $\mathcal{V} \setminus (\mathcal{B} \cup \mathcal{C})$ perceive the collusion as independent players with a distinct strategy each, whereas in reality they are all subject to a single strategy dictated by the controlling entity, Eve.

Theorem 4 (Sybil Resilience).
Let $\mathcal{G}$ be a game graph and $(\mathcal{B}, \mathcal{C})$ be a collusion of players on $\mathcal{G}$. It holds that

$$Tr_{A \to \mathcal{B} \cup \mathcal{C}} = Tr_{A \to \mathcal{B}} \ . \tag{47}$$

Proof Sketch. The incoming trust to $\mathcal{B} \cup \mathcal{C}$ cannot be higher than the incoming trust to $\mathcal{B}$ since $\mathcal{C}$ has no incoming trust from players outside the collusion. For the complete proof, see Appendix (6.3). □

We have proven that controlling $|\mathcal{C}|$ is irrelevant for Eve, thus Sybil attacks are meaningless.

5 Related Work

6 Further Research

While our trust network can form a basis for risk-invariant transactions in the anonymous and decentralized setting, more research is required to achieve other desirable properties. Some directions for future research are outlined below.

6.1 Zero knowledge

Our network evaluates indirect trust by computing the max flow in the graph of lines-of-credit. In order to do that, complete information about the network is required. However, disclosing the network topology may be undesirable, as it subverts the identity of the participants even when participants are treated pseudonymously, as deanonymisation techniques can be used [13]. To avoid such issues, exploring the ability to calculate flows in a zero knowledge fashion may be desirable. However, performing network queries in zero knowledge may allow an adversary to extract topological information. More research is required to establish how flows can be calculated effectively in zero knowledge and what bounds exist in regards to information revealed in such fashion.

The current description of TrustIsRisk refers to a static setting where the game evolves in turns. In each turn only one user changes the state

of the network. In the dynamic setting, the users should be able to play simultaneously, freely join, leave and disconnect temporarily from the network.

Appendix

6.2 Common Notation

Definition 19 (Assets of A in turn j, $As_{A,j}$).
Sum of A's capital and outgoing trust.

$$As_{A,j} = Cap_{A,j} + out_{A,j} \tag{48}$$

If the turn we refer to is obvious, it is possible to omit j.

Let $A = Player(j)$. $Turn_j$ Examples:

1.
$$Turn_j = \emptyset \tag{49}$$

2.
$$Turn_j = \{Steal\,(y, B)\,, Add\,(w, B)\}\ , \tag{50}$$

given that

$$DTr_{B\to A,j_2-1} \leq y \wedge -DTr_{A\to B,j_2-1} \leq w \wedge y - w \leq Cap_{A,j_2-1}\ . \tag{51}$$

3.
$$Turn_j = \{Steal\,(x, B)\,, Add\,(y, C)\,, Add\,(w, D)\}\ , \tag{52}$$

given that

$$\begin{gathered} DTr_{B\to A,j_3-1} \leq x \wedge -DTr_{A\to C,j_3-1} \leq y \wedge \\ \wedge -DTr_{A\to D,j_3-1} \leq w \wedge x - y - w \leq Cap_{A,j_3-1}\ . \end{gathered} \tag{53}$$

4.
$$Turn_j = \{Steal\,(x, B)\,, Steal\,(y, B)\} \tag{54}$$

is not a valid turn because it contains two $Steal\,()$ actions against the same player. If

$$x + y \leq DTr_{B\to A}\ , \tag{55}$$

the correct alternative would be

$$Turn_j = \{Steal\,(x + y, B)\}\ . \tag{56}$$

6.3 Proofs, Lemmas and Theorems

Lemma 3 (*Loss* Equivalent to *Damage*).
Let $j \in \mathbb{N}, v \in \mathcal{V}_j \setminus \{A, E\}, v = Player\,(j)$. *It holds that*

$$\min\left(in_{v,j}, Loss_{v,j}\right) = \min\left(in_{v,j}, Damage_{v,j}\right) \quad . \tag{57}$$

Proof. – Let $v \in Happy_{j-1}$. Then

1. $v \in Happy_j$ because $Turn_j = \emptyset$,
2. $Loss_{v,j} = 0$ because otherwise $v \notin Happy_j$,
3. $Damage_{v,j} = 0$, or else any reduction in direct trust to v would increase equally $Loss_{v,j}$ (line 16), which cannot be decreased again but during an Angry player's turn (line 18).
4. $in_{v,j} \geq 0$

Thus

$$v \in Happy_{j-1} \Rightarrow \min\left(in_{v,j}, Damage_{v,j}\right) = \min\left(in_{v,j}, Loss_{v,j}\right) = 0 \quad . \tag{58}$$

– Let $v \in Sad_{j-1}$. Then

1. $v \in Sad_j$ because $Turn_j = \emptyset$,
2. $in_{v,j} = 0$ (lines 27 - 28),
3. $Damage_{v,j} \geq 0 \wedge Loss_{v,j} \geq 0$.

Thus

$$v \in Sad_{j-1} \Rightarrow \min\left(in_{v,j}, Damage_{v,j}\right) = \min\left(in_{v,j}, Loss_{v,j}\right) = 0 \quad . \tag{59}$$

– Let $v \in Angry_{j-1} \wedge v \in Happy_j$. Then the same argument as in the first case holds, if we ignore the argument (1).
– Let $v \in Angry_{j-1} \wedge v \in Sad_j$. Then the same argument as in the second case holds, if we ignore the argument (1).
Thus the theorem holds in every case. □

Proof (Trust Convergence Theorem (1)). First of all,

$$\forall j > j_0 : Player\,(j) = E \Rightarrow Turn_j = \emptyset \tag{60}$$

because E has already nullified his incoming and outgoing direct trusts in $Turn_{j_0}$, the evil strategy does not contain any case where direct trust is increased or where the evil player starts directly trusting another player and the other players do not follow a strategy in which they can choose to $Add\,()$ trust to E, thus player E can do nothing $\forall j > j_0$. We also see that

$$\forall j > j_0 : Player(j) = A \Rightarrow Turn_j = \emptyset \tag{61}$$

because of the idle strategy that A follows. As far as the rest of the players are concerned, consider the Transitive Game. As we can see from lines 5 and 16 - 18, it is

$$\forall j, \sum_{v \in \mathcal{V}_j} Loss_v = in_{E,j_0-1} \ . \tag{62}$$

In other words, the total loss is constant and equal to the total value stolen by E. Also, as we can see in lines 3 and 28, which are the only lines where the Sad set is modified, once a player enters the Sad set, it is impossible to exit from this set. Also, we can see that players in $Sad \cup Happy$ always pass their turn. We will now show that eventually the $Angry$ set will be empty, or equivalently that eventually every player will pass their turn. Suppose that it is possible to have an infinite amount of turns in which players do not choose to pass. We know that the number of nodes is finite, thus this is possible only if

$$\exists j' : \forall j \geq j', |Angry_j \cup Happy_j| = c > 0 \wedge Angry_j \neq \emptyset \ . \tag{63}$$

This statement is valid because the total number of angry and happy players cannot increase because no player leaves the Sad set and if it were to be decreased, it would eventually reach 0. Since $Angry_j \neq \emptyset$, a player v that will not pass her turn will eventually be chosen to play. According to the Transitive Game, v will either deplete her incoming trust and enter the Sad set (line 28), which is contradicting $|Angry_j \cup Happy_j| = c$, or will steal enough value to enter the $Happy$ set, that is v will achieve $Loss_{v,j} = 0$. Suppose that she has stolen m players. They, in their turn, will steal total value at least equal to the value stolen by v (since they cannot go sad, as explained above). However, this means that, since the total value being stolen will never be reduced and the turns this will happen are infinite, the players must steal an infinite amount of value, which is impossible because the direct trusts are finite in number and in value. More precisely, let j_1 be a turn in which a conservative player is chosen and

$$\forall j \in \mathbb{N}, DTr_j = \sum_{w,w' \in \mathcal{V}} DTr_{w \to w',j} \ . \tag{64}$$

Also, without loss of generality, suppose that

$$\forall j \geq j_1, out_{A,j} = out_{A,j_1} \ . \tag{65}$$

In $Turn_{j_1}$, v steals

$$St = \sum_{i=1}^{m} y_i \ . \tag{66}$$

We will show using induction that

$$\forall n \in \mathbb{N}, \exists j_n \in \mathbb{N} : DTr_{j_n} \leq DTr_{j_1-1} - nSt \ . \tag{67}$$

Base case: It holds that

$$DTr_{j_1} = DTr_{j_1-1} - St \ . \tag{68}$$

Eventually there is a turn j_2 when every player in $N^-(v)$ will have played. Then it holds that

$$DTr_{j_2} \leq DTr_{j_1} - St = DTr_{j_1-1} - 2St \ , \tag{69}$$

since all players in $N^-(v)$ follow the conservative strategy, except for A, who will not have been stolen anything due to the supposition.
Induction hypothesis: Suppose that

$$\exists k > 1 : j_k > j_{k-1} > j_1 \Rightarrow DTr_{j_k} \leq DTr_{j_{k-1}} - St \ . \tag{70}$$

Induction step: There exists a subset of the $Angry$ players, S, that have been stolen at least value St in total between the turns j_{k-1} and j_k, thus there exists a turn j_{k+1} such that all players in S will have played and thus

$$DTr_{j_{k+1}} \leq DTr_{j_k} - St \ . \tag{71}$$

We have proven by induction that

$$\forall n \in \mathbb{N}, \exists j_n \in \mathbb{N} : DTr_{j_n} \leq DTr_{j_1-1} - nSt \ . \tag{72}$$

However

$$DTr_{j_1-1} \geq 0 \wedge St > 0 \ , \tag{73}$$

thus

$$\exists n' \in \mathbb{N} : n'St > DTr_{j_1-1} \Rightarrow DTr_{j_{n'}} < 0 \ . \tag{74}$$

We have a contradiction because

$$\forall w, w' \in \mathcal{V}, \forall j \in \mathbb{N}, DTr_{w \to w', j} \geq 0 \ , \tag{75}$$

thus eventually $Angry = \emptyset$ and everybody passes. □

Proof (MaxFlows Are Transitive Games Lemma (1)).
Without loss of generality, we suppose that the interesting turn is 0. In other words, $j_0 = 0$. Let $X = \{x_{vw}\}_{\mathcal{V}\times\mathcal{V}}$ be the flows returned by the execution of the $MaxFlow\,(A, B)$ algorithm on $\mathcal{G}_0$. It is known that for

any directed weighted graph G there exists a $MaxFlow$ over G that is a DAG [citation needed]. We also know that we can apply the topological sort algorithm to any DAG and obtain a total ordering of its nodes with the following property: $\forall$ nodes v, w, it holds that $v < w \Rightarrow x_{wv} = 0$ [citation needed]. Put differently, there is no flow from larger to smaller nodes. We execute the topological sort on X and obtain a total order of the nodes, such that B is the maximum and A is the minimum node. B is maximum since it is the sink and thus has no outgoing flow to any node and A is minimum since it is the source and thus has no incoming flow from any node. The desired execution of algorithm 4 will choose players following the total order, starting from player B. We observe that $\forall v \in \mathcal{V} \setminus \{A, B\}, \sum\limits_{w \in \mathcal{V}} x_{wv} = \sum\limits_{w \in \mathcal{V}} x_{vw} \leq maxFlow\left(A, B\right) \leq in_{B,0}$. Player B will follow a modified evil strategy where she steals value equal to her total incoming flow, not her total incoming trust. Let j_2 be the first turn when A is chosen to play. We will show using strong induction that there exists a set of valid actions for each player according to their respective strategy such that at the end of each turn j the corresponding player $v = Player\left(j\right)$ will have stolen value x_{wv} from each in neighbour w.
Base case: In turn 1, B steals value equal to $\sum\limits_{w \in \mathcal{V}} x_{wB}$, following the modified evil strategy.

$$Turn_1 = \bigcup_{v \in N^-(B)_0} \{Steal\left(x_{vB}, v\right)\} \tag{76}$$

Induction hypothesis: Let $k \in [j_2 - 2]$. We suppose that $\forall j \in [k]$, there exists a valid set of actions, $Turn_j$, performed by $v = Player\left(j\right)$ such that v steals from each player w value equal to x_{wv}.

$$\forall j \in [k], Turn_j = \bigcup_{w \in N^-(v)_{j-1}} \{Steal\left(x_{wv}, w\right)\} \tag{77}$$

Induction step: Let $j = k + 1, v = Player\left(j\right)$. Since all the players that are greater than v in the total order have already played and all of them have stolen value equal to their incoming flow, we deduce that v has been stolen value equal to $\sum\limits_{w \in N^+(v)_{j-1}} x_{vw}$. Since it is the first time v plays, $\forall w \in N^-\left(v\right), DTr_{w \to v, j-1} = DTr_{w \to v, 0} \geq x_{wv}$, thus v is able to choose the following turn:

$$Turn_j = \bigcup_{w \in N^-(v)_{j-1}} \{Steal\left(x_{wv}, w\right)\} \tag{78}$$

Moreover, this turn satisfies the conservative strategy since

$$\sum_{w \in N^-(v)_{j-1}} x_{wv} = \sum_{w \in N^+(v)_{j-1}} x_{vw} \ . \tag{79}$$

Thus $Turn_j$ is a valid turn for the conservative player v.
We have proven that in the end of turn $j_2 - 1$, player B and all the conservative players will have stolen value exactly equal to their total incoming flow, thus A will have been stolen value equal to her outgoing flow, which is $maxFlow(A, B)$. Since there remains no Angry player, it is obvious that j_2 is a turn that Transitive Game has converged thus $Loss_{A,j_2} = Loss_A$. It is also obvious that if B had chosen the original evil strategy, the described actions would still be valid only by supplementing them with additional $Steal\,()$ actions, thus $Loss_A$ would further increase. This proves the theorem. □

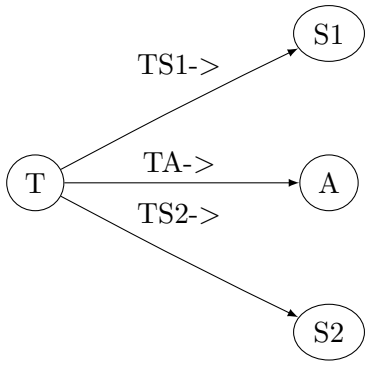
Proof (Transitive Games Are Flows Lemma (2)). Let $Sad, Happy, Angry$ be as defined in the Transitive Game. Let $\mathcal{G}'$ be a directed weighted graph based on $\mathcal{G}$ with an auxiliary source. Let also j_1 be a turn when the Transitive Game has converged. More precisely, $\mathcal{G}'$ is defined as follows:

$$\mathcal{V}' = \mathcal{V} \cup \{T\} \tag{80}$$

$$\mathcal{E}' = \mathcal{E} \cup \{(T, A)\} \cup \{(T, v) : v \in Sad_{j_1}\} \tag{81}$$

$$\forall (v, w) \in \mathcal{E}, c'_{vw} = DTr_{v \to w,0} - DTr_{v \to w,j_1} \tag{82}$$

$$\forall v \in Sad_{j_1}, c'_{Tv} = c'_{TA} = \infty \tag{83}$$



We observe that $\forall v \in \mathcal{V}$,

$$\sum_{w \in N^-(v) \setminus \{T\}} c'_{wv} = \sum_{w \in N^-(v) \setminus \{T\}} (DTr_{w \to v,0} - DTr_{w \to v,j_1}) = in_{v,0} - in_{v,j_1} \tag{84}$$

and

$$\sum_{w \in N^+(v) \setminus \{T\}} c'_{vw} = \sum_{w \in N^+(v) \setminus \{T\}} (DTr_{v \to w,0} - DTr_{v \to w,j_1}) = out_{v,0} - out_{v,j_1} \ . \tag{85}$$

We can suppose that

$$\forall j \in \mathbb{N}, in_{A,j} = 0 \ , \tag{86}$$

since if we find a valid flow under this assumption, the flow will still be valid for the original graph.
Next we try to calculate $MaxFlow\,(T, B) = X'$ on graph $\mathcal{G}'$. We observe that a flow in which it holds that $\forall v, w \in \mathcal{V}, x'_{vw} = c'_{vw}$ can be valid for the following reasons:

- $\forall v, w \in \mathcal{V}, x'_{vw} \leq c'_{vw}$ (Capacity flow requirement (36) $\forall e \in \mathcal{E}$)
- Since $\forall v \in Sad_{j_1} \cup \{A\}, c'_{Tv} = \infty$, requirement (36) holds for any flow $x'_{Tv} \geq 0$.
- Let $v \in \mathcal{V}' \setminus (Sad_{j_1} \cup \{T, A, B\})$. According to the conservative strategy and since $v \notin Sad_{j_1}$, it holds that

  $$out_{v,0} - out_{v,j_1} = in_{v,0} - in_{v,j_1} \ . \tag{87}$$

  Combining this observation with (84) and (85), we have that

  $$\sum_{w \in \mathcal{V}'} c'_{vw} = \sum_{w \in \mathcal{V}'} c'_{wv} \ . \tag{88}$$

  (Flow Conservation requirement (37) $\forall v \in \mathcal{V}' \setminus (Sad_{j_1} \cup \{T, A, B\})$)
- Let $v \in Sad_{j_1}$. Since v is sad, we know that

  $$out_{v,0} - out_{v,j_1} > in_{v,0} - in_{v,j_1} \ . \tag{89}$$

  Since $c'_{Tv} = \infty$, we can set

  $$x'_{Tv} = (out_{v,0} - out_{v,j_1}) - (in_{v,0} - in_{v,j_1}) \ . \tag{90}$$

  In this way, we have

  $$\sum_{w \in \mathcal{V}'} x'_{vw} = out_{v,0} - out_{v,j_1} \text{ and} \tag{91}$$

  $$\sum_{w \in \mathcal{V}'} x'_{wv} = \sum_{w \in \mathcal{V}' \setminus \{T\}} c'_{wv} + x'_{Tv} = in_{v,0} - in_{v,j_1} + \\ + (out_{v,0} - out_{v,j_1}) - (in_{v,0} - in_{v,j_1}) = out_{v,0} - out_{v,j_1} \ . \tag{92}$$

  thus

  $$\sum_{w \in \mathcal{V}'} x'_{vw} = \sum_{w \in \mathcal{V}'} x'_{wv} \ . \tag{93}$$

  (Requirement 37 $\forall v \in Sad_{j_1}$)

– We set

$$x'_{TA} = \sum_{v \in \mathcal{V}'} x'_{Av} \ , \tag{94}$$

thus from (86) we have

$$\sum_{v \in \mathcal{V}'} x'_{vA} = \sum_{v \in \mathcal{V}'} x'_{Av} \ . \tag{95}$$

(Requirement 37 for A)

We saw that for all nodes, the necessary properties for a flow to be valid hold and thus X' is a valid flow for $\mathcal{G}$. Moreover, this flow is equal to $maxFlow(T, B)$ because all incoming flows to B are saturated. Also we observe that

$$\sum_{v \in \mathcal{V}'} x'_{Av} = \sum_{v \in \mathcal{V}'} c'_{Av} = out_{A,0} - out_{A,j_1} = Loss_A \ . \tag{96}$$

We define another graph, $\mathcal{G}''$, based on $\mathcal{G}'$.

$$\mathcal{V}'' = \mathcal{V}' \tag{97}$$

$$E(\mathcal{G}'') = E(\mathcal{G}') \setminus \{(T, v) : v \in Sad_j\} \tag{98}$$

$$\forall e \in E(\mathcal{G}''), c''_e = c'_e \tag{99}$$

If we execute the algorithm $MaxFlow(T, B)$ on the graph $\mathcal{G}''$, we will obtain a flow X'' in which

$$\sum_{v \in \mathcal{V}''} x''_{Tv} = x''_{TA} = \sum_{v \in \mathcal{V}''} x''_{Av} \ . \tag{100}$$

The outgoing flow from A in X'' will remain the same as in X' for two reasons:
No capacity reachable by A is modified and T has no incoming flow, thus

$$\sum_{v \in \mathcal{V}''} x''_{Av} \geq \sum_{v \in \mathcal{V}'} x'_{Av} \text{[citation needed] and} \tag{101}$$

$$\left.\begin{array}{l} \sum\limits_{v \in \mathcal{V}''} c''_{Av} = \sum\limits_{v \in \mathcal{V}'} c'_{Av} = \sum\limits_{v \in \mathcal{V}'} x'_{Av} \\ \sum\limits_{v \in \mathcal{V}''} c''_{Av} \geq \sum\limits_{v \in \mathcal{V}''} x''_{Av} \end{array}\right\} \Rightarrow \sum_{v \in \mathcal{V}''} x''_{Av} \leq \sum_{v \in \mathcal{V}'} x'_{Av} \tag{102}$$

Thus we conclude that

$$\sum_{v \in \mathcal{V}''} x''_{Av} = \sum_{v \in \mathcal{V}'} x'_{Av} \ . \tag{103}$$

Let $X = X'' \setminus \{(T, A)\}$. Observe that

$$\sum_{v \in \mathcal{V}''} x''_{Av} = \sum_{v \in \mathcal{V}} x_{Av} \ . \tag{104}$$

This flow is valid on graph $\mathcal{G}$ because

$$\forall e \in \mathcal{E}, c_e \geq c''_e \ . \tag{105}$$

Thus there exists a valid flow for each execution of the Transitive Game such that

$$\sum_{v \in \mathcal{V}} x_{Av} = \sum_{v \in \mathcal{V}''} x''_{Av} \overset{(103)}{=} \sum_{v \in \mathcal{V}'} x'_{Av} \overset{(96)}{=} Loss_{A,j_1} \ , \tag{106}$$

which is the flow X. □

Theorem 5 (Conservative World Theorem).
If everybody follows the conservative strategy, nobody steals any amount from anybody.

Proof Sketch. If everybody is conservative, nobody can initiate the chain of steals. □

Proof.
Suppose that we are interested in graphs $\mathcal{G}_j$.

$$\begin{gathered} \text{Let } (j_k) \text{ an increasing sequence of positive integers,} \\ \text{let } S_{j_k} \subseteq N^- \left(Player\left(j_k\right)\right)_{j_k} \text{ and} \\ \text{let } \forall v \in S_{j_k}, y_{v,j_k} > 0 \ . \end{gathered} \tag{107}$$

Suppose that there exists a subseries of History, $(Turn_{j_k})$, where

$$Turn_{j_k} = \bigcup_{v \in S_{j_k}} \{Steal(y_{v,j_k}, v)\} \ , \tag{108}$$

This subseries must have an initial element, $Turn_{j_1}$. However, $Player(j_1)$ follows the conservative strategy, thus somebody must have stolen from her as well, so $Player(j_1)$ cannot be the initial element. We have a contradiction, thus the theorem holds. □

Proof (Sybil Resilience Theorem (4)). Let $\mathcal{G}_1$ be a game graph defined as follows:

$$\mathcal{V}_1 = \mathcal{V} \cup \{T_1\} \ , \tag{109}$$

$$\mathcal{E}_1 = \mathcal{E} \cup \{(v, T_1) : v \in \mathcal{B} \cup \mathcal{C}\} \ , \tag{110}$$

$$\forall v, w \in \mathcal{V}_1 \setminus \{T_1\}, DTr^1_{v\to w} = DTr_{v\to w} \ , \tag{111}$$

$$\forall v \in \mathcal{B} \cup \mathcal{C}, DTr^1_{v\to T_1} = \infty \ , \tag{112}$$

where $DTr_{v\to w}$ is the direct trust from v to w in $\mathcal{G}$ and $DTr^1_{v\to w}$ is the direct trust from v to w in $\mathcal{G}_1$.
Let also $\mathcal{G}_2$ be the induced graph that results from $\mathcal{G}_1$ if we remove the Sybil set, $\mathcal{C}$. We rename T_1 to T_2 to facilitate comprehension. (Image)
According to /heorem (3),

$$Tr_{A\to\mathcal{B}\cup\mathcal{C}} = maxFlow_1(A, T_1) \wedge Tr_{A\to\mathcal{B}} = maxFlow_2(A, T_2) \ . \tag{113}$$

We will show that the $MaxFlow$ of each of the two graphs can be used to construct a valid flow of equal value for the other graph. The flow $X_1 = MaxFlow(A, T_1)$ can be used to construct a valid flow of equal value for the second graph if we set

$$\forall v \in \mathcal{V}_2 \setminus \mathcal{B}, \forall w \in \mathcal{V}_2, x_{vw,2} = x_{vw,1} \ , \tag{114}$$

$$\forall v \in \mathcal{B}, x_{vT_2,2} = \sum_{w\in N^+(v)} x_{vw,1} \ , \tag{115}$$

$$\forall v, w \in \mathcal{B}, x_{vw,2} = 0 \ . \tag{116}$$

Therefore

$$maxFlow_1(A, T_1) \leq maxFlow_2(A, T_2) \tag{117}$$

Likewise, the flow $X_2 = MaxFlow(A, T_2)$ is a valid flow for $\mathcal{G}_1$ because $\mathcal{G}_2$ is an induced subgraph of $\mathcal{G}_1$. Therefore

$$maxFlow_1(A, T_1) \geq maxFlow_2(A, T_2) \tag{118}$$

We conclude that

$$maxFlow(A, T_1) = maxFlow(A, T_2) \ , \tag{119}$$

thus from (113) and (119) the theorem holds. □

References

1. Patrikakis C., Masikos M., Zouraraki O.: Distributed Denial of Service Attacks. The Internet Protocol Journal, Vol. 7, N. 4, http://www.cisco.com/c/en/us/about/press/internet-protocol-journal/back-issues/table-contents-30/dos-attacks.html (2004)
2. Standard ebay selling fees, http://pages.ebay.com/help/sell/fees.html
3. ebay money back guarantees, http://pages.ebay.com/ebay-money-back-guarantee/questions.html
4. What is OpenBazaar, https://blog.openbazaar.org/what-is-openbazaar/
5. Can Bitcoin and Multisig Reduce Identity Theft and Fraud?, https://blog.openbazaar.org/can-bitcoin-and-multisig-reduce-identity-theft-and-fraud/
6. Nakamoto S.: Bitcoin: A Peer-to-Peer Electronic Cash System (2008)
7. Bitcoin Developer Guide, https://bitcoin.org/en/developer-guide
8. Kahn Arthu.r B.: Topological sorting of large networks. Communications of the ACM Vol. 5, Issue 11, pp. 558-562, ACM, New York (1962)
9. Cormen T. H., Leiserson C. E., Rivest R. L., Stein C.: Introduction to Algorithms (3rd ed.). MIT Press and McGraw-Hill (2009) [1990]
10. Zindros D. S.: Trust in decentralized anonymous marketplaces (2015)
11. Buterin V.: Bitcoin Multisig Wallet: The Future of Bitcoin. Bitcoin Magazine (2014), https://bitcoinmagazine.com/articles/multisig-future-bitcoin-1394686504
12. Sanchez W.: Lines of Credit (2016) https://gist.github.com/drwasho/2c40b91e169f55988618#part-3-web-of-credit
13. Narayanan A., Shmatikov V.: De-anonymizing Social Networks. SP '09 Proceedings of the 2009 30th IEEE Symposium on Security and Privacy, pp. 173-187, 10.1109/SP.2009.22 (2009)