A Trustless Privacy-Preserving Reputation System

Alexander Schaub, Rémi Bazin, Omar Hasan, Lionel Brunie

A trustless privacy-preserving reputation system

Alexander Schaub[1], Rémi Bazin[1], Omar Hasan[2], and Lionel Brunie[2]

[1] Ecole polytechnique, 91128 Palaiseau, France
[2] University of Lyon, CNRS, INSA-Lyon, LIRIS, UMR5205, F-69621, France

Abstract. Reputation systems are crucial for distributed applications in which users have to be made accountable for their actions, such as e-commerce websites. However, existing systems often disclose the identity of the raters, which might deter honest users from submitting reviews out of fear of retaliation from the ratees. While many privacy-preserving reputation systems have been proposed, we observe that none of them is simultaneously truly decentralized, trustless, and suitable for real world usage in, for example, e-commerce applications. In this paper, we present a blockchain based decentralized privacy-preserving reputation system. We demonstrate that our system provides correctness and security while eliminating the need for users to trust any third parties or even fellow users.

1 Introduction

These days, reputation systems are implemented in various websites, where they are crucial for the customer experience. For instance, buyers are inclined to pay more for goods if the seller has a good reputation [1]. One of the first and best-studied systems in the e-commerce domain is the reputation system at `ebay.com` [2]. Its main objective is to help prospective customers to determine the trustworthiness of the sellers, and thus minimize the risk of fraud.

A study [2] showed that users may retaliate in case of negative feedback, and thus raters are less likely to provide honest feedback. In order to avoid this problem, several privacy preserving solutions have been proposed. Some of them try to hide the identity of the ratee [3,4,5], while others try to hide the rating [6,7,8] while making the aggregated reputation public.

While some of the existing privacy preserving reputation systems might be suitable for e-commerce applications, we observe that each one of them comes with its drawbacks. For example, Kerschbaum's system [9] has been specifically designed with e-commerce in mind. However, it is a centralized system, and thus can potentially be abused by the central authority. Other schemes [8] achieve anonymity even in this context, but are not trustless.

Given these considerations, we would like to achieve a *trustless reputation system*, i.e. one that does not require the participants to trust other users or entities to not disrupt the protocol or to breach their privacy. This privacy-preserving reputation model should be suitable for e-commerce applications, and

we will therefore suppose that the identity of the customer is revealed during the transactions that they can rate.

In order to achieve true trustlessness we also require our system to be decentralized. One way to obtain decentralization is to use a distributed database in order to store the ratings submitted by the customers. We will achieve this using *blockchains*.

The blockchain technology, which became popular thanks to the BitCoin protocol [10], has been used in various applications. Among these applications, we can count a domain name system (DNS) named Namecoin. The blockchain can be more generally, as explained in [11], seen as a public distributed database, with all the participants agreeing about its state in a secure manner. In BitCoin, for example, this database serves to store a ledger of the coins that each user owns, as well as the transactions between the users.

Anonymous reputation systems are a natural application for the blockchain technology. There have already been some attempts at building such systems [12], however, there seems to be no usable solution yet.

We will leverage this technology in order to achieve the objectives of our reputation system. It will enable us to build a truly decentralized system, that does not require the participants to trust other users, as the integrity of the rating-history can be verified by every user.

We propose a truly trustless, decentralized, anonymity preserving reputation system that is suitable for e-commerce applications. It is based on the blockchain technology, and will induce low overhead for the processing of transactions, while at the same time be robust and allow customers to submit ratings as well as textual reviews.

The rest of the paper is organized as follows. In Section 2, we will analyze existing privacy-preserving systems and explain in further detail why they are not suitable for e-commerce applications. In Section 3, we will explain the model used for our system, and list the properties that we want to achieve. Then, in Section 4, we will describe the necessary building blocks, and in Section 5, we will present our system in detail. Finally, we will explain in Section 6, why this system meets the expected goals. We conclude the paper in Section 7.

2 Related Work

Privacy preserving reputation systems have been studied in the literature for a long time. One of the first proposed systems was designed by Pavlov et al. [6] and uses primitives such as the secure sum and verifiable secret sharing. It protects the confidentiality of the feedback by hiding the values of the submitted ratings. Hasan et al. [8] later introduced a system based on additive homomorphic cryptography and Zero-Knowledge proofs where the privacy of a given user can be preserved even in the presence of a large majority of malicious users. A

little later, Dimitriou et al. [7] proposed two protocols with a similar architecture to the systems presented by Hasan et al., with slightly higher asymptotic complexity, however, less demanding in terms of resources for the querier (he has to relay less messages, verify less proofs, etc.).

Some protocols [6,8,7,13,14] are truly decentralized and the feedback is retrieved from the participants every time a querier wishes to learn the reputation of another participant. Therefore, all the nodes have to stay online in order to contribute to the reputation system, which is not suitable for e-commerce applications, but might be useful in other contexts, such as P2P applications.

Hence, we will focus on privacy-preserving methods that completely hide the identity of the raters. Protocols of such type do already exist, however each one of them has its own weaknesses. The works of Androulaki et al. [13] and Petrlic et al. [14], for example, are instances of pseudonym based schemes. Nonetheless, these two require a Trusted Third Party (TTP), and are thus not truly decentralized. As the TTP has to be completely trusted for certain operations, its misbehavior could breach the privacy of the users or the correctness of the system.

Anceaume et al. [3,4] proposed slightly different solutions. Instead of all the information about the reputation of the users being held by a single TTP, they distribute the trust using a DHT-structure: every peer holds some part of the information, which allows to compute the reputation of a service provider. Moreover, in their system, peers rate *transactions* between customers and service providers, rather than directly rating service providers. This seems more suitable for e-commerce applications, as one would typically rate every transaction made with a service provider, rather than periodically update their opinion on a given service provider. It also allows to introduce proofs of transactions, which guarantee (more or less) that only transactions that really took place can be rated. However, as the service provider creates those proofs, it is complicated to ensure that he doesn't generate proofs for transactions that did not happen, in order to submit positive reviews by himself and wrongfully increase his own reputation. Anceaume's and Lajoie-Mazenc's systems only offer little protection against these attacks. The system proposed in [4] also makes use of complicated zero-knowledge proofs and is thus quite costly to perform (several seconds for each participant, up to a minute in certain cases).

None of those protocols are trustless, and therefore need either the customers or the service providers (or both) to trust some entities not to tamper with the system or to break privacy, without being able to verify that there is no bad behavior. We eliminate this weakness in the system that we present in this paper.

3 Our model

3.1 Participants

For our system, we choose a model that is as close as possible to actual e-commerce systems. As stated in Section 1, we will consider two types of users :

service providers (SP) who will sell goods or services, and customers who might buy them. The most important part in e-commerce rating systems is the rating of the service providers (as opposed to ratings from the seller about the buyer). Therefore, we will only consider ratings *from* the customers *about* the SPs. Only customers might be raters, and only SP will be ratees.

We will also suppose that the transaction will ***disclose*** the identity of the customer : the SP will need the customer's credentials, such as his credit card number or address, in order to process the order. Even if the transaction is done via an anonymous electronic currency such as `Dashcoin` or `Zerocash`, the service provider will most certainly need the customer's address in order to deliver the good. We suppose that after every transaction between a customer and a SP, the customer might rate the SP.

More formally, we will introduce the following notations :

S : The set of all the service providers (i.e. ratees)

C : The set of all the customers (i.e. raters)

P : The set of all the participants, $P := S \cup C$. It is simply the set of all the nodes participating in the network.

B : The blockchain.

As the blockchain defines an ordered set of blocks. A block is simply a set of operations that are aggregated for maintenance reasons (it is more efficient to store them this way). The blockchain can also be seen as a database whose state will be the initial state (that is hard-coded) on which all the operations contained in the subsequent blocks are applied.
Every time a new block is constituted, an award will be paid to the user that constituted it. This works in a similar fashion as in the so-called "alt-coins". In our system, owning coins is mandatory in order to be allowed to receive reputation. It also helps preventing spam and other kinds of attacks (as described in section 6.2).

A : The set of all the addresses of the participants.

These addresses will be used for maintenance. Every service provider will own one address. They will be used, in particular, to hold and spend the coins generated by the blockchain, but also to identify the service providers.

Service providers will have a unique address, as issuing reputation tokens will cost coins and owning an address is necessary in order to own and transfer them. As a service provider will not gain anything from having more than one address (see section 6.2 for more details), there is no need to try and enforce this policy. Furthermore, it would be complex to enforce it in a decentralized fashion.

3.2 Operations

We will next describe the functions that are needed in our system. The protocols that implement these functions will be described in the later sections. Most, if

not all, of these functions will be performed with respect to a given blockchain B, or need to make calls to a random number generator (RNG). These are implicit inputs of the protocols.

For the customer These operations will be performed by a certain customer $c \in C$.

- setup()
  Generates a new public key, usable by the customer, for the transaction.
- get_reputation(s)
  Allows the customer to query the reputation of a service provider $s \in S$.
- get_token(s, x)
  Allows the customer to request a token that will prove that he was engaged in a transaction x with the service provider s. Outputs a blinded token $\bar{t}_x$.
- unblind_token($\bar{t}_x$)
  Unblinds the token that was retrieved using the `get_token` protocol. Outputs an unblinded token t_x. The token is bound to the transaction. However, given two tokens t_x and $t_{x'}$, the service provider s will not be able to tell which token belongs to which transaction.
- publish_review(s, t_x)
  Allows the customer to publish a review about the service provider $s \in S$, using the token t_x previously unblinded.

For the service provider This operation will be performed by the service provider.

- issue_token(c, x)
  In response to a `get_token` request from the customer $c \in C$, issues a blinded token $\bar{t}$ and sends it to the customer, if the customer is entitled to receive one, i.e. he was really engaged in the transaction x.

Block-chain related operations These operations are mostly independent from the underlying reputation model. However, a blockchain mechanism is needed in order to store the reputation values in a reliable way. These operations can be performed by any node in the network.

- broadcast(op)
  Broadcasts an operation op (which can be a review, a transaction, etc.) to all the nodes running the protocol.
- compute_balance(s)
  For a service provider $s \in S$, representing an address $addr$, computes the balance (in terms of coins) associated with this service provider.
- create_new_block($addr, b$)
  Broadcasts a newly mined block b. It will contain, among other data, reviews and transactions, but also a proof that $addr$ has the right to constitute the next block.

The incentive for creating blocks are the *coins*. Every address who correctly creates a block and broadcasts it will receive a certain amount of coins. They serve as a currency within this system, in a similar fashion as in many cryptocurrencies that have been developed since BitCoin. However, in our system, the main usage of the coins is *not* to serve as an alternative currency. Rather, they will be needed by the service providers in order to have the right to deliver tokens. This also serves to limit spam from the service providers, who could simply create as many tokens as they desire in order to boost their reputation.

3.3 Adversarial model

We consider a malicious adversarial model with collusions. This model implies that any participant in the protocol may behave arbitrarily and deviate from the protocol at any time as deemed necessary. Service providers may want to learn the identity of the customers that rated them, they might try to raise their own reputation, and collaborate with other service providers. Customers may try to submit reviews without having previously interacted with service providers, might try to use the received token in order to rate other service providers, or might try to otherwise disrupt the service. We will also suppose that there might be attempts to disrupt the blockchain, such as forking in order to confuse new participants.

3.4 Objectives

The objectives for our system are the following :

- *Trustlessness*
  In an e-commerce system, we cannot expect customers to have pre-existing trust towards other customers of the same SP. Therefore, our system should not suppose that there is pre-existing subjective trust between users. No customer should have to trust any entity *not* to deviate from the protocol in order to break its privacy or change its rating. The protocol should ensure that privacy and correctness are preserved even if other participants deviate from the protocol. Therefore, it should also not rely on Trusted Third Parties, or Certification Authorities, which, by definition, must be trusted to behave faithfully.
- *Suitability for e-commerce*
  As the identity of a customer will be most certainly revealed during a transaction, the system should enforce the unlinkability of transactions and ratings, i.e. for a given rating, it should not be possible to determine which transaction it is related to (it should however be possible to identify the related SP). It should, however, not be possible for a customer to submit a rating if no transaction took place.
- *Decentralization*
  We want to avoid any central point of failure as well as any single point of control. Therefore, the system should not depend on one, two, or a small

number of nodes in order to work properly. We will even exclude Certification Authorities, because they have proven unreliable in the past, either because they became subject to attacks [15,16] or because they issued themselves fraudulent certificates [17], and because they would induce some centralization aspects in the system.
- *Anonymity preservation*
  The anonymity of the customers should be preserved. More precisely, the ratings and the identities of the customers should be unlinkable, as well as the ratings among themselves. The later kind of unlinkability is also crucial to preserve the anonymity of the users, as highlighted in [18] and [19].
- *Robustness*
  Our system should be robust to classical attacks on reputation systems, in particular bad-mouthing, ballot-stuffing, Sybil attacks and whitewashing.

4 Building blocks

In order to be able to build this protocol, we will need two basic building blocks : the blockchain and blind signatures.

4.1 Blockchain

A blockchain, as first desribed in [10], can be seen as a distributed, public database, which can be read by every user running the appropriate program, but on which writing has a cost, or cannot be done at any time by any user ([11]). Every action that modifies this database is broadcasted among all the users in the network, and they are recorded as "blocks". The creation of those "blocks" is controlled by mechanisms that vary between the different blockchain algorithms, and the state of the database is the sum of all the actions in all the blocks at a given moment in time. This concept has become popular due to the BitCoin currency [10], which seems to be the first application making use of this idea. Two families of blockchain systems have since then emerged.

The first one uses a mechanism for controlling the blockchain that is similar to the one used in BitCoin, in which the probability of a participant creating a new block is proportional to its computing power. The second one uses a different mechanism, in which the amounts of coins held by the participant define this probability. This is called Proof-of-Stake, and we advocate the use of such a blockchain system for our protocol. More information about the different blockchain systems can be found in the extended version of this paper [20].

4.2 Blind signatures

A blind signature scheme is a protocol in which the signer of a message does not learn anything about the content of the message that was signed. We expect from such a system the following properties :

Unforgeablility
The signature cannot be falsified (only the user knowing some secret information, such as a private key, can issue valid signatures).

Blindness
The signer does not learn anything about the message it signs (given the information available to the signer, all possible messages are equally likely to be about to be signed).

Untraceability
Once the message and signature have been revealed, the signer cannot determine when the message was signed.

For example, the blind signature scheme proposed by Okamoto [21], based on bilinear pairings, could be used to instantiate this primitive, or the simpler version based on the RSA algorithm, first proposed by Chaum [22]. As Chaum's version is simpler and faster, we will use this scheme in order to explain our protocol.

5 Specification of the protocol

5.1 An Overview

The proposed protocol could be summarized as follows :

1. Before contacting the service provider in order to perform a transaction, the customer may compute the service provider's reputation using the get_reputation protocol.
2. Once the customer retrieved the reputation of the service provider, he decides whether to engage in a transaction with the SP or not.
3. If the customer decides to engage in a transaction, before a transaction takes place, the customer creates a new *public key*, derived from a private/public key pair, for the process. This key should be kept secret from the service provider for the moment. It will be used to avoid token-theft Then, the transaction takes place : for example, the customer sends the money, and the SP starts to deliver the good.
4. Just after the transaction takes place, the customer asks the SP for a *blinded token* by performing the get_token protocol (which takes the freshly generated public key as input), and verifies that the SP has a sufficient balance for issuing a token (using the compute_balance protocol). The balance should be greater than some n coins, since n coins will be deduced from the SP's account when the review will be integrated in the blockchain. The customer then verifies the token (i.e. verifies that the signature is correct) and unblinds it for later use with help of the unblind_token protocol.

   Requiring some coins to be spent in order to receive a review helps to prevent ballot-stuffing attacks, as the SPs may, theoretically, issue an unlimited amount of tokens to themselves and could therefore submit an unlimited

number of positive reviews for themselves. As for the token, it serves as a proof that a transaction really occurred. It therefore helps to greatly reduce the risk of bad mouthing attacks. It has to be blinded, so that the service provider cannot link the token, and therefore the rating, to the transaction and the identity of the customer.

5. Once the customer is ready to review the SP, he will broadcast a message containing the address of the SP, the token, along with the rating of the transaction and (optionally) a written review, a signature on this information, as well as a pointer to the last review concerning the same service provider. This is done via the publish_review protocol. a cash system.
6. A participant wishing to earn coins (in order to be allowed to grant tokens for example) verifies if he is allowed to constitute the next block. If it is the case, he will run the constitute_block protocol. The creation of blocks helps to maintain a unique history of actions, avoids double-use of tokens, and is incentivized through the reward in coins.

In the next sub-sections, we will describe the protocol in more detail.

5.2 Public key creation

Before the transaction takes place, the customer creates a new public key that will be used for one transaction only (similar to what is recommended for BitCoin addresses for example). This will be the public part of an ECDSA key [23]. It must not be communicated to the SP during the setup phase. An outline of the setup protocol could look like follows :

Algorithm 1 Setup protocol

1: **procedure** SETUP(T)
2: $(p, a, b, G, n, h) \leftarrow T$ // Those are the parameters of the elliptic curve used for ECDSA, for example those of secp256k1
3: $privKey \leftarrow rand(0, n)$
4: $pubKey \leftarrow privKey * G$
return $(pubKey, privKey)$

5.3 Blinded token exchange

Before the transaction takes place, the customer will receive a token from the SP that will guarantee that its review will be accepted. For this purpose, the customer hashes the previously generated public key and requests a blind signature on this, for example using Okamoto's provable blind signature scheme (in the complete, not partial blinding setup), or the much simpler Chaum's blind signature algorithm. This will make the token unlinkable to the transaction, and therefore guarantee the anonymity. The customer will also check that there are

enough coins in the wallet associated with the SP. Then, the transaction can take place.

If Chaum's blind signature is used, then we can define the three protocols get_token(s), unblind_token($\bar{t}$) and issue_token(c) as in **Algorithm 2**.

Algorithm 2 Token exchange

procedure GET_TOKEN($s, pubKey, e, n, x$) ▷ (e, n) is the service provider's public RSA key pair, x the identifier of the transaction
  $m_1 \leftarrow hash(pubKey)$ ▷ $hash$ is a cryptographic hash function such as sha256
  $r \leftarrow rand(0, n)$
  $m_1 \leftarrow m_1 r^e \bmod n$
  $send((m_1, x), s)$
  return (m_1, r)
procedure ISSUE_TOKEN(c, m_1, d, n, x) ▷ (n, d) is the service provider's private RSA key pair
  if verify(x) **then** ▷ the service provider has to specify verify
    $\bar{t} \leftarrow m_1^d \bmod n$
    $send(\bar{t}, c)$
    return $\bar{t}$
procedure UNBLIND_TOKEN($\bar{t}, r, e, n$)
  $t \leftarrow \bar{t} r^{-1} \bmod n$
  return t

We suppose that the customer knows the public key of the SP. How this is done is out of scope of this paper.

5.4 Broadcasting the review

Once the transaction is finished, the customer might want to wait for some time (so that he is not the SP's only customer for this period). After this period of waiting, he might choose a rating for this transaction (say, an integer in $[|0; 5|]$) and write a review about it. The review can give helpful information to prospective customers, explain a bad rating, and helps distinguishing between trustworthy and fake ratings. This information will be broadcast in the network, along with the identifier of the SP, the token and the signature on the token. This message will also contain the signature of the customer, and a pointer to the last review concerning the service provider. The message that will be broadcast can be represented as follows :

5.5 Computing the reputation

In order to compute the reputation, a new customer only needs the last block containing a review about the SP whose reputation it seeks. Once this block has been found, it is sufficient to follow the pointers in order to retrieve all the reviews about this SP. For each review, the prospective customer might also verify the correctness of the blinded tokens. Then, the customer can choose any aggregation

Table 1: Structure of a broadcasted message containing a review

Field	Description
$addr_s$	Address of the SP
$pubKey$	The public key used for the transaction
$token$	Token obtained from the SP (i.e. blind signature on $pubKey$)
$rating$	Rating of the transaction (for example, an integer in $[0; 5]$)
$review$	A textual review on the transaction (optional)
sig	Signature, using $privKey$, of $(adds_s \| \| pubKey \| \| token \| \| rating \| \| review)$
$pointer$	Pointer to the last review about the same SP

function he wishes (mean, median, or beta-reputation [24]), and could also read the textual reviews in order to filter out outlier ratings (especially high or, more probably, especially low ratings).

6 Analysis of the protocol

6.1 Security Analysis

In this section, we list the theorems whose proof demonstrates that we achieve the security objectives of our protocol. The proofs to the theorems are quite straight forward and can be found in the extended version of the paper [20].

Theorem 1 (Token unforgeability). *Given a Service Provider's public key, and a poly-bounded (in a security parameter k) number of signatures from the Service Provider on arbitrary messages, a user is not able to generate one more token (i.e. signature on the hash of an address) except with negligible probability $\epsilon(k)$.*

Remark 1. This implies that badmouthing attacks are not possible on this system. Ballot-stuffing attacks, however, cannot be completely mitigated, as a service provider can freely issue tokens. The currency introduced in this protocol helps reducing the risk of ballot stuffing, as the service provider is limited on the number of tokens he can issue, by the number of coins he owns.

Theorem 2 (Reputation unforgeability). *Given the public blockchain history, no service provider is able to advertise a reputation that is not its own (except with negligible probability $\epsilon(k)$).*

Theorem 3 (Customer anonymity). *Given a rating published in the blockchain concerning a given service provider, the identity from the customer that originated this rating is indistinguishable, from the service provider's point of view, among all the customers that were previously involved in a transaction with that service provider.*

Theorem 4 (Customer report unlinkability). *Given two different reports, it is not possible to determine whether they were issued by the same customer or not better than guessing at random.*

The proposed protocol is able to hide the identity of the rater among all the customers who interacted with a given service provider in a certain time interval, therefore providing indistinguishability. We can devise a simple model that will give an idea of the indistinguishability of the customer reviews. Suppose that customers purchase goods from a given service provider. The arrival of customers can be modeled by a Poisson process with parameter λ. We can also suppose that, after receiving their good, customers will wait for a certain amount of time before submitting a review. In our model, the customer will wait for a time T that is uniformly distributed over $[0; \tau]$ for some $\tau > 0$. In this case, we have the following property :

Theorem 5 (Indistinguishability). *If the arrival of customers is modeled as a Poisson process of parameter λ and if customers wait for a duration that is uniformly distributed over $[0; \tau]$ before submitting their review, then the identity of a customer will be indistinguishable over a set of $\lambda\tau$ customers in average.*

6.2 Robustness against generic attacks

In this section, we will explain how our proposed system copes with generic attacks against reputation systems : bad-mouthing, ballot stuffing, Sybil attacks, and whitewashing.

Bad-mouthing Bad-mouthing consists in lying about the performance of a service provider in order to decrease his reputation. This could be done, for example, by a competitor. Our system prevents bad-mouthing thanks to the usage of **tokens**, and this attack is prevented because token unforgeability is guaranteed by the system.

Ballot-stuffing Ballot stuffing is the opposite of bad-mouthing. This attack consists in increasing one's own reputation. As the service providers generate the tokens that allow feedback-submission on their own, this attack could only partially be mitigated with the use of **coins**, as explained in the remark concerning token unforgeability.

Whitewashing Whitewashing consists in exiting a system after having accumulated bad reputation, in order to re-enter it again and removing the accumulated bad reputation. As the initial reputation of a new service provider is 0, the service provider would not gain much from leaving and re-entering the system with a new identity. However, bad reviews are worse than no reviews, so there could be an incentive in order to do so. One way to limit this would be to bind the identity of a service provider to, for example, his website, through a specific operation on the blockchain. The service provider could still change the domain name, but again, this would cost money.

Sybil attacks Sybil attacks combine more or less the attacks described above. They consist in creating multiple identities in the system in order to disrupt it. They pose no more threat than the other types of attacks, as there is no concept of "identity" for the customers in our system, and creating multiple identities for a service provider can only be used to either perform whitewashing (if he creates one identity after another) or ballot-stuffing (if he creates multiple fake transactions).

7 Conclusion

Reputation systems need to be privacy-preserving in order to work properly, without the raters having to be afraid of retaliation. Building a reputation system that is privacy-preserving without any trust assumptions is not a trivial task. However, such a system would be highly valuable, because there is much less risk that the privacy of the users could be breached. We described such a reputation system for e-commerce applications, and analyzed the security guarantees. Some points would still need attention in future work, such as the exact way of generating coins that would ensure that service providers have enough of them in order to be able to supply enough tokens for their customers, but at the same time still limit ballot-stuffing attacks. Also, we must find a definite way to address the problem of information leakage concerning the time at which the reviews are submitted.

References

1. Paul Resnick, Richard Zeckhauser, John Swanson, and Kate Lockwood. The value of reputation on ebay: A controlled experiment. *Experimental Economics*, 9(2):79–101.
2. Paul Resnick and Richard Zeckhauser. Trust among strangers in internet transactions: Empirical analysis of ebays reputation system. *The Economics of the Internet and E-commerce*, 11(2):23–25, 2002.
3. Emmanuelle Anceaume, Gilles Guette, Paul Lajoie Mazenc, Nicolas Prigent, and Valérie Viet Triem Tong. A Privacy Preserving Distributed Reputation Mechanism. October 2012.
4. Paul Lajoie-Mazenc, Emmanuelle Anceaume, Gilles Guette, Thomas Sirvent, and Valérie Viet Triem Tong. Efficient Distributed Privacy-Preserving Reputation Mechanism Handling Non-Monotonic Ratings. January 2015.
5. John Bethencourt, Elaine Shi, and Dawn Song. Signatures of reputation. In *Proceedings of the 14th International Conference on Financial Cryptography and Data Security*, FC'10, pages 400–407, Berlin, Heidelberg, 2010. Springer-Verlag.
6. Elan Pavlov, JeffreyS. Rosenschein, and Zvi Topol. Supporting privacy in decentralized additive reputation systems. In Christian Jensen, Stefan Poslad, and Theo Dimitrakos, editors, *Trust Management*, volume 2995 of *Lecture Notes in Computer Science*, pages 108–119. Springer Berlin Heidelberg, 2004.
7. Tassos Dimitriou and Antonis Michalas. Multi-party trust computation in decentralized environments in the presence of malicious adversaries. *Ad Hoc Netw.*, 15:53–66, April 2014.

8. Omar Hasan, Lionel Brunie, Elisa Bertino, and Ning Shang. A decentralized privacy preserving reputation protocol for the malicious adversarial model. *IEEE Transactions on Information Forensics and Security*, 8(6):949–962, 2013.
9. Florian Kerschbaum. A verifiable, centralized, coercion-free reputation system. In *Proceedings of the 8th ACM Workshop on Privacy in the Electronic Society*, WPES '09, pages 61–70, New York, NY, USA, 2009. ACM.
10. Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2008. `https://bitcoin.org/bitcoin.pdf`.
11. Marc Pilkington. Blockchain technology: Principles and applications. *Research Handbook on Digital Transformations, edited by F. Xavier Olleros and Majlinda Zhegu. Edward Elgar*, 2016.
12. Davide Carboni. Feedback based reputation on top of the bitcoin blockchain. *arXiv preprint arXiv:1502.01504*, 2015.
13. Elli Androulaki, Seung Geol Choi, Steven M. Bellovin, and Tal Malkin. Reputation systems for anonymous networks. In *Proceedings of the 8th International Symposium on Privacy Enhancing Technologies*, PETS '08, pages 202–218, Berlin, Heidelberg, 2008. Springer-Verlag.
14. Ronald Petrlic, Sascha Lutters, and Christoph Sorge. Privacy-preserving reputation management. In *Proceedings of the 29th Annual ACM Symposium on Applied Computing*, SAC '14, pages 1712–1718, New York, NY, USA, 2014. ACM.
15. Comodo report of incident - comodo detected and thwarted an intrusion on 26-mar-2011. `https://www.comodo.com/Comodo-Fraud-Incident-2011-03-23.html`.
16. Key internet operator verisign hit by hackers. `http://www.reuters.com/article/2012/02/02/us-hacking-verisign-idUSTRE8110Z820120202`.
17. Maintaining digital certificate security. `http://googleonlinesecurity.blogspot.fr/2015/03/maintaining-digital-certificate-security.html`.
18. A. Narayanan and V. Shmatikov. Robust de-anonymization of large sparse datasets. In *Security and Privacy, 2008. SP 2008. IEEE Symposium on*, pages 111–125, May 2008.
19. Michael Barbaro and Tom Zeller Jr. A face is exposed for aol searcher no. 4417749, August 2006.
20. Alexander Schaub, Rmi Bazin, Omar Hasan, and Lionel Brunie. A trustless privacy-preserving reputation system. Cryptology ePrint Archive, Report 2016/016, 2016. `http://eprint.iacr.org/`.
21. Tatsuaki Okamoto. Efficient blind and partially blind signatures without random oracles. In *Theory of cryptography*, pages 80–99. Springer, 2006.
22. David Chaum, Amos Fiat, and Moni Naor. Untraceable electronic cash. In *Proceedings on Advances in cryptology*, pages 319–327. Springer-Verlag New York, Inc., 1990.
23. Don Johnson, Alfred Menezes, and Scott Vanstone. The elliptic curve digital signature algorithm (ecdsa). *International Journal of Information Security*, 1(1):36–63, 2001.
24. Audun Jøsang and Roslan Ismail. The beta reputation system. In *Proceedings of the 15th bled electronic commerce conference*, volume 5, pages 2502–2511, 2002.