

Trust Arrays: Allowing P2P nodes to “personally” evaluate trustworthiness of potential partners

Mauro Stocco, Thomas Engel, Uwe Roth
Université du Luxembourg
Faculté des Sciences,
de la Technologie et de la Communication
Email: {mauro.stocco, thomas.engel, uwe.roth}@uni.lu

Abstract—Managing Trust in peer-to-peer (P2P) environments is an issue of particular importance. The adaptation of the meaning of the word “Trust” to P2P systems is already complicated on the level of its own definition. Current trust evaluation models, like user evaluation on eBay, other web auction systems or online shops, cannot be applied. Since they rely on central data storage systems, these models are not scalable. This paper gives a global view on different tries to evaluate trust in a P2P context and to introduce a simple approach to manage the trustworthiness of the participants in a P2P network.

I. INTRODUCTION

Besides the huge commercial success of different P2P systems in the last years, which we do not need to explain in further detail, the idea of being able to share information and resources with people all around the world gained a huge interest in the research communities, as this is the ideology behind any academical research. In the last years, a lot of research was done in order to develop “pure” [1] peer-to-peer information systems, not relying on central servers, like Chord [2], CAN [3], Pastry [4], Tapestry [5] or Kademlia [6].

The results of these researches also lead to many possible current and new applications that could rely on distributed systems based on P2P networks. As an example, members of the Chord project published a work on supporting DNS upon Chord [7], though making any root server unnecessary. Another highly discussed topic is the use of decentralized P2P networks for web caching [8]. P2P networks could also become the platform for many other network based applications like telephony, videoconferencing or any other kind of streaming [9].

The main part of this research was realized by a detailed study of the different publications on the named P2P protocols and resulted on a more detailed study of Chord [2].

A fundamental problem in peer-to-peer applications is an effective location of the node containing the desired data item using a **decentralized** architecture, though not being dependent on a centralized indexing system that represents a single point of failure. This is what the Chord protocol is all about. Given a key, it maps this key onto a node. The main (and only) issue of the Chord protocol is this keying problem.

The data keys are distributed on the Chord Ring. To build up a Chord Ring, the consistent hash function assigns each node and each key an m -bit identifier using SHA-1 [10]. The

value of m must be chosen large enough to make collisions improbable, though to make it improbable that two nodes get the same key if the *id space* is too small. The SHA-1 hash of the key string returns a *key Identifier*. The SHA-1 hash of a node’s IP-Address and its virtual node identifier returns the *node Identifier* in the flat key space. Note that both types of identifiers are placed on the same *identifier space* namely the *Chord Ring*.

The Chord Ring is an identifier circle modulo 2^m . All identifiers are ordered on the Chord Ring according to the following rule. Key k is assigned to the first node whose identifier is equal or larger than k . This node is called the *successor* of k .

The key location is done using a scalable key location algorithm that allows to find a given key within $O(\log N)$ requests (with N , the total number of nodes, currently in the network). This protocol also perfectly deals with frequent architecture changes, meaning many nodes joining and leaving continuously, which is the core idea of P2P networks.

II. TRUST MANAGEMENT IN P2P NETWORKS - WHAT IS TRUST ?

With the evolution of P2P networks and electronic business in the last years, the importance of trust management became more and more crucial in electronic communities. In online auction systems (ex. eBay), sellers and bidders can evaluate each other. In different online shops (ex. Amazon), customers can rate their product in order to help future customers in their decision. Other interesting examples are online forums, where a user’s reputation grows with the amount of posts but can also be influenced by moderators, who may judge the user’s behaviour as incorrect and punish him therefore by giving some negative points. All these reputation systems are completely centralized and rely on a central data management system (i.e. database, LDAP server).

In P2P networks, peers must find out if they can trust each other. A peer must know whom it is talking to and if the communicating peer is really the one it is trying to be held for. That is just the identity problematics. Furthermore, a peer must know if the identified peer can provide trustworthy information. A trustworthy peer will fairly participate to the network by routing correctly and sharing correct and uncompromising information or reliable resources. Furthermore, trust must also

be context aware. Depending on what information a peer wants to have, it may trust more a certain peer and have no trust at all in another one that may be trustworthy in a different context. This is a straightforward comparison to real life as you do not go to a plumber to get your car fixed. This is a very largely discussed and current topic as many publications witness [11]–[14].

As we are talking of P2P networks, the management of these trust issues must be completely decentralized, though all reputation data must not rely on any kind of central system like a database or another kind of central server. In [15], the authors give a game-theoretic framework for analysing trust in different decentralized protocols.

III. HOW TO MANAGE TRUST IN A P2P ENVIRONMENT

The main problem that comes up when trying to manage trust in “pure” P2P networks is that the information about the trustworthiness of different agents will be spread around the network and it is not always available since agents (peers) are not reachable all the time. This means that, when trying to gather information about a potential partner, a peer will only be able to obtain a subset of the available information.

Another issue is that one cannot blindly trust other peers when they provide information about trustworthiness of other agents. They could belong to a subset of peers trying to gain trust in order to plan an attack afterwards.

The reputation system could be based on experiences a node or his neighbours already made with other nodes. A node wanting to communicate with another node will first check if it already knows the target node and if this is not true, try to gather information from its other neighbours on the concerned node. The node who wants to get information on his potential collaborator must find out if the information other nodes try to give him is trustworthy and true. If this is not given, the information on the other peer’s reputation is worthless. Furthermore, there must be a kind of fair mechanism to give a chance to new nodes with no reputation yet to participate in order to build up their own reputation. This is important for load balancing too as otherwise, in a very short time, some nodes would be overloaded while others would not be allowed to participate.

Aberer and Despotovic [17] proposed a decentralized trust management system based on the transactions between peers. Only complaints due to cheating of the partner peer are kept in mind. This system also proposes a data replication mechanism in order to keep information on different peers. The data is stored using P-Grid [18], a peer-to-peer lookup system based on a virtual distributed search tree, which was also published by Aberer. Other decentralized systems could be studied too for this approach.

Another recent trust management algorithm is the Eigen-Trust Algorithm [19] which judges the trust level of a peer according to its recent upload history.

IV. TRUST ARRAYS

There are many different possibilities to store reputation data over one specific node. The idea in this work was to store

information about different parameters separately. This has the advantage that when a node is looking for a partner node, it can choose it according to the criterion it judges as most important. One node may put more importance on the quality of the transmission link while the other might prefer to communicate with partners that only had successful transactions until now.

A. Proposed Data Structure

This paper would like to introduce *Trust Arrays* as a simple data structure allowing to store and transmit some parameters about a node’s trustworthiness. These parameters are chosen in order to allow any peer, independently of its age in the network to gain reputation. Furthermore, any peer reading the information from a Trust Array can independently decide which information to consider and what priority to give it. Any node can also define its own threshold value that a parameter must have to be positively considered. This means that having the same value in a Trust Array concerning one certain node, a more critical peer would refuse communication with this peer while another one could perfectly accept it.

In order to communicate Trust Array values, nodes should be correctly identified to one another, for example, using a decentralized PKI architecture like explained in [20] or any other mechanism to ensure that the information comes from the location we are expecting it from. The identification of the peer does not give information on the trustworthiness of the peer but at least, the information concerning the Trust Array will not be falsified on its way. If a node has the Trust Array entries from different sources, it can calculate them and decide its own trust policy in order to chose a peer.

A trustworthy peer will fairly participate to the network by routing correctly and sharing correct and uncompromising information or reliable resources.

B. Trust Array Values

In this section, the values that could be part of such a Trust Array are proposed. Most of them should be expressed in a format without absolute numbers. This prevents having to cope with too large numbers. If a value is expressed as coefficient, it allows nodes that are new in the network to build up a reputation from the very beginning of their participation in the network. Otherwise in a very short time, a network relying on absolute trust parameters would only consider a very small subset of nodes whose reputation would grow rapidly leading these nodes to be overloaded while others would be excluded.

Proposed values to introduce in a Trust Array :

- The **node Id** is a universally unique identifier (UUID) and must be contained in the array as any node can store information about any other node in its array. The node Id though gives certainty on which node is concerned.
- The amount of **successful transactions** will be counted as follows. For every successful transaction, the counter is incremented, for every complaint by another node, it is decremented. The counter would result in an absolute number. In order not to get such a giant number to compare and for fairness purposes, one could also convert

this coefficient in percent. This would have the advantage that every node's reputation would rely on successful transactions in relation to the total amount of transactions. Using this kind of coefficient, a new node will not have to proceed to a huge amount of transactions before getting an acceptable reputation.

- The **diversity coefficient** will be calculated from the number of transactions to different peers compared to all the transactions. As an example, if node n has a total number of x transactions to z different peers, its diversity coefficient is z/x . This will always be a positive number lesser than 1 and can easily be expressed in percent.
- The **availability on the network**, can be calculated in percent per day or per hour. In most cases this will be a pretty low value as we know that in peer-to-peer networks, presence is not very high.
- A **timestamp** should also be part of the array in order to check whether information coming from another node can still be considered as up to date.

C. Trust Calculation

If a node wants to communicate with another one, it must find out if the level of trust of the target node is high enough. On the other hand, the target node must check if the node, it is being contacted from, is trustworthy or if it better had to refuse the communication. So before establishing a connection, nodes must check their local table and ask their reachable neighbours whether the partner is trustworthy or not. As the nodes will never be able to reach a very large number of other nodes in the network, an interesting approach is to study to which extend the "small world problem" [16] can be applied to trust evaluation.

According to the proposed structure for the Trust Array, the data that the nodes acquire is defined, but the priority the nodes give to any part of the array and the minimum value to accept is the decision of every node. Though it is absolutely possible that a more critical node would refuse communication where another one would accept it.

The values of the trust array should be introduced somewhere on application level in a simple daemon running in the background and recalculating the values when a new entry for a given node is communicated.

V. CONCLUSIONS

The aim of this paper was to discuss the different possible values to evaluate trust in form of the proposed array. We plan to enhance the model as much as possible before bringing it in an application environment. The trust values proposed are objective in the sense that they do not have anything to do with the "personality" of a node. The proposed values express the node's participation in a network. The format and the evaluation mechanism of the different values in the array allow every single node to evaluate another node in its own way by setting own priorities and thresholds on the different values. This allows the evaluating node a certain part of subjectivity in the evaluation part.

Trust Arrays are a quite simple data structure allowing peers to follow a common guideline for the exchange of trust values, still letting every participant the whole liberty to decide upon which criterion to judge a potential partner. This makes the calculation of the trust level a completely independent operation for every peer. This means that the rules to apply for the calculation of trustworthiness completely underly to the decision of the different peers. As this idea of Trust Arrays is still on its very beginning, its implementation and application will show to which extend such a system is realizable.

VI. FUTURE WORK

Future work will consist in trying to implement a trust evaluation system relying on Trust Arrays upon an existing P2P protocol (eg. Chord). Concrete mappings and definitions will have to be developed for trust calculation with the possibility to easily exchange them if a more performant model comes up. Using this implementation, tests and simulations will show to which extend such a method is realizable. The other issue will be to test how the identification of the nodes will have to be realized in order to be usable.

ACKNOWLEDGMENT

This work is funded by the SECAN-Lab at the University of Luxembourg.

REFERENCES

- [1] R. Schollmeier, "A Definition of Peer-to-Peer Networking for the Classification of Peer-to-Peer Architectures and Applications," *First International Conference on Peer-to-Peer Computing (P2P'01)*, 2001. [Online]. Available: <http://csdl.computer.org/comp/proceedings/p2p/2001/1503/00/15030101.pdf>
- [2] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," in *Proceedings of the 2001 conference on applications, technologies, architectures, and protocols for computer communications*. ACM Press, 2001, pp. 149–160. [Online]. Available: 1-58113-411-8
- [3] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A scalable content addressable network," Berkeley, CA, Tech. Rep. TR-00-010, 2000. [Online]. Available: citeseer.ist.psu.edu/ratnasamy01scalable.html
- [4] A. Rowstron and P. Druschel, "Pastry: Scalable, Decentralized Object Location, and Routing for Large-Scale Peer-to-Peer Systems," *Lecture Notes in Computer Science*, vol. 2218, pp. 329–350, 2001. [Online]. Available: citeseer.ist.psu.edu/rowstron01pastry.html
- [5] K. Hildrum, J. D. Kubiawicz, S. Rao, and B. Y. Zhao, "Distributed Object Location in a Dynamic Network," in *Proceedings of the Fourteenth ACM Symposium on Parallel Algorithms and Architectures*, aug 2002, pp. 41–52. [Online]. Available: citeseer.ist.psu.edu/article/hildrum02distributed.html
- [6] P. Maymounkov and D. Mazières, "Kademlia: A peer-to-peer information system based on the xor metric," 2002. [Online]. Available: citeseer.ist.psu.edu/article/maymounkov02kademlia.html
- [7] R. Cox, A. Muthitacharoen, and R. Morris, "Serving DNS using a peer-to-peer lookup service," 2002. [Online]. Available: R. Cox, A. Muthitacharoen, and R. T. Morris, Serving DNS using a Peer-to-Peer Lookup Service, in IPTPS, Mar. 2002.
- [8] S. Iyer, A. Rowstron, and P. Druschel, "Squirrel: A decentralized peer-to-peer web cache," 2002. [Online]. Available: citeseer.ist.psu.edu/iyer02squirrel.html
- [9] P. Chou, V. Padmanabhan, and H. Wang, "Resilient peer-to-peer streaming," 2003. [Online]. Available: citeseer.ist.psu.edu/padmanabhan03resilient.html
- [10] P. E. J. C. S. Donald E. Eastlake (Motorola), "US Secure Hash Algorithm 1(SHA1)," The Internet Engineering Task Force, RFC 3174, September 2001, <http://www.ietf.org/rfc/rfc3174.txt>.

- [11] W. Y. R. Chen, "Poblano - a distributed trust model for peer-to-peer networks," <http://security.jxta.org>.
- [12] F. Cornelli, E. Damiani, and S. D. Capitani, "Choosing reputable servers in a p2p network," 2002. [Online]. Available: citeseer.ist.psu.edu/556137.html
- [13] E. Damiani, S. di Vimercati, S. Paraboschi, P. Samarati, and F. Violante, "A reputation based approach for choosing reliable resources in peer-to-peer networks," 2002. [Online]. Available: citeseer.ist.psu.edu/damiani02reputationbased.html
- [14] S. A. Singh, "TrustMe: Anonymous Management of Trust Relationships in Decentralized P2P," 2003. [Online]. Available: Aameek Singh and Ling Liu. TrustMe: Anonymous Management of Trust Relationships in Decentralized P2P Systems. In IEEE Intl. Conf. on Peer-to-Peer Computing, Sep. 2003.
- [15] T.-I. P. Ruggero, "A game-theoretic framework for analyzing." [Online]. Available: citeseer.ist.psu.edu/687738.html
- [16] H. Zhang, A. Goel, and R. Govindan, "Using the Small-World Model to Improve Freenet Performance," 2002. [Online]. Available: citeseer.ist.psu.edu/zhang02using.html
- [17] K. Aberer and Z. Despotovic, "Managing trust in a Peer-2-Peer information system," in *Proceedings of the Tenth International Conference on Information and Knowledge Management (CIKM01)*, H. Paques, L. Liu, and D. Grossman, Eds. ACM Press, 2001, pp. 310–317. [Online]. Available: citeseer.ist.psu.edu/abarer01managing.html
- [18] K. Aberer, "P-Grid: A self-organizing access structure for P2P information systems," *Lecture Notes in Computer Science*, vol. 2172, pp. 179–??, 2001. [Online]. Available: citeseer.ist.psu.edu/abarer01pgrid.html
- [19] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina, "The EigenTrust Algorithm for Reputation Management in P2P Networks," in *Proceedings of the Twelfth International World Wide Web Conference (WWW)*, 2003. [Online]. Available: citeseer.ist.psu.edu/kamvar03eigentrust.html
- [20] A. Datta, M. Hauswirth, and K. Aberer, "Beyond 'Web of Trust': Enabling P2P e-commerce." [Online]. Available: citeseer.ist.psu.edu/article/datta03beyond.html
- [21] F. Stajano and R. Anderson, "The resurrecting duckling: Security issues for ad-hoc wireless networks," in *Security Protocols, 7th International Workshop Proceedings*, 1999, pp. 172–194. [Online]. Available: citeseer.ist.psu.edu/article/stajano99resurrecting.html
- [22] B. Lampson, M. Abadi, M. Burrows, and E. Wobber, "Authentication in distributed systems: Theory and practice," *ACM Transactions on Computer Systems*, vol. 10, no. 4, pp. 265–310, 1992. [Online]. Available: citeseer.ist.psu.edu/article/lampson92authentication.html
- [23] M. Abadi, M. Burrows, B. Lampson, and G. Plotkin, "A calculus for access control in distributed systems," *ACM Transactions on Programming Languages and Systems*, vol. 15, no. 4, pp. 706–734, September 1993. [Online]. Available: citeseer.ist.psu.edu/article/abadi91calculus.html