Reputation System for Large Scale Environments

Roman Špánek
Institute of Computer Science, Academy of Sciences of the Czech Republic
Pod Vodárenskou věží 2
Czech Republic
roman.spanek@tul.cz
Technical University of Liberec
Hálkova 6
Czech Republic

Abstract

The paper describes a new approach for treating trust in reconfigurable groups of users with special accent on trust in the next generations of the Internet. The proposed model uses properties of weighted hypergraphs. Model flexibility enables description of relations between nodes such that these relations are preserved under frequent changes. The ideas can be straightforwardly generalized to other concepts describable by weighted hypergraphs. The consistency of the proposal was verified in a couple of experiments with our pilot implementation SecGRID.

1. Introduction

The Semantic Web is widely believed to be the successor of the current web. Its main idea is to describe resources in the form of machine processable meta-data allowing automation of the requested tasks connected with the retrieval and usage of these resources. Although the main focus of previous work was aimed at the creation of knowledge representation languages (RDF-S, DAML+OIL, OWL), reasoning systems, and also at the tools helping to embed web pages with semantic markup, the emerging commercial applications such as e-commerce, banking or travel services face a lot of security issues.

Unfortunately, current security mechanisms used in distributed systems (e.g. Kerberos [20], PGP [8], SPKI [7], etc.) cannot be seamlessly transferred to the Internet due to extremely large number of resources, services, agents and users, their heterogeneity, and the rapidity of changes in its structure.

Therefore the main goal of the paper is to study, propose and verify a trust management system for large scale distributed environments.

The paper is organized as follows. Next Section shortly summarizes related work followed by the introduction of the novel trust model. In the next sections 3.2,3.3 we describe creation of basic structure of entities and dynamic evolution of the structure, respectively. Section 4 presents experimental results related to the verification of the stability and section 5 concludes the paper.

2 Related Work

2.1 Trust Management

Trust management systems can be categorized as follow:

- *credential* and *policy based trust management*;
- *reputation based trust management*, and;
- *social network based trust management*.

This categorization is based upon the way adopted for establishing and evaluating trust between entities.

Policy based approach has been proposed in the context of open and distributed services architectures [3], [16], [2], [4] as well as in the context of Grids [1] as the solution to the problem of authorization and access control in open systems. Its focus is on the trust management mechanisms employing different policy languages and engines for specifying and reasoning on rules for the trust establishment. Since the primary aim of such systems is to enable access control, trust management is limited to verification of credentials and restricting access to resources according to policies defined by required resources owner [10].

On the contrary, *Reputation based* trust management systems provide a way in which entities may evaluate and build a trust relationship between resource provider and requester. Reputation approach emerged in the context of electronic commerce systems, e.g. eBay. In distributed settings, reputation-based approaches have been proposed for managing trust in public key certificates, P2P systems XREP, mobile ad-hoc networks, and recently, also in the Semantic Web [5], NICE [15], DCRC/CORC [11], EigenTrust [13], [14], [6], [12], [21].

Social network based trust management systems utilize, in addition, social relationships between entities to infer trust. In particular, the social network based system views the whole structure as a social network with relationships defined amongst entities. Examples of such trust management systems include Regret [19], NodeRanking [17].

2.2 Dynamic Trust Management

The reputation systems have been shown to be appropriate for maintaining trust in decentralized systems. Trust can be, beside the other applications, used for access control in many distributed environments with little or no centralized control. Nevertheless the trust in P2P, mobile databases, the semantic web as well as in the real human society is highly dynamic.

In most dynamic approaches trust is defined as a vector comprising few factors contributing to the overall trust value (e.g. [6]):

- the *short term trust factor*,
- the *long term trust factor*,
- the *penalty factor*.

These factors are then combined into one value of *dynamic trust metric* of a particular connection between entities. The purpose of the factors can be generalized as an effort to accommodate sudden deviation in normal behavior of an entity (so-called oscillation) together with long term behavior observation. The penalty factor is concerned to make reaction of the system (decrease or increase of trust level) satisfactory.

3 Proposal of the Novel Trust Model

Our model exceeds the known models for building trust in *interpretation* of the trust. The other proposals consider each entity as individual being *responsible for its own relationships*. Even the reputation systems where a indirect trust can be inferred, consider entities as individuals. The same can be asserted for dynamic trust models as well as the other models (etc. PGP[8], PKI[7]).

In our point of view, the trust level is common for a group of users rather than individuals. In this way trust is not a single value for particular two entities but rather shared value for a set (group) of entities. As the groups can differ in purpose[1], one entity can be member of more groups. Trust between two entities is than inferred based on their groups memberships. Such model allows building of trust between mutually unknown entities easily with less communication and computation load.

We consider such trust interpretation and treatment highly useful especially in the Internet, where amount of users is growing day by day, and where traditional trust management systems face severe efficiency problems.

3.1 The Model

This section gives a brief overview on the security model. For details readers are referred to our foregoing papers [18],[22].

Currently available solutions concern concrete trust between members where a direct relationship exists or where a transitive relationship can be found. Such approaches can be very naturally modeled as oriented weighted graphs where vertices correspond to members and edges represent relationships (either direct or indirect path). Trust is simply described by weights of edges.

The graph model is sufficient in the case of complicated relationships among members, but modeling groups of users with demanded efficiency could pose some problems. A group of users can be modeled as a graph where group members are connected by edges with the same weights, but such approach suffers by space load overhead.

In hypergraph $\mathcal{H} = (U, N, W_U, W_N)$, on the other hand, a hyperdge can connect arbitrary many vertices[2] and one vertex can be a pin of more hyperedges [9]. In our hypergraph model the following relations hold:

- vertices U represent users
- the weight of vertice W_{u_i} represents user u_i related information (abilities, etc.)
- hyperedges N represent groups of users
- the weight of hyperedge W_{n_i} represents overall group security together with group related information
- pins of a hyperedge $pins(n_i)$ represent the members of the group described by the hyperedge n_i
- $hyperedges(u)$ represents set of groups of a user u

Figures 3.1 and 3.1 show simple examples of representation of groups of users as a graph and hypergraph. The interesting situation arises between users G,H,I,J,D,C in

[1] There might be groups of tennis players as well as lawyers, researchers, musicians, friends,...

[2] The upper bound is naturally given by the amount of vertices $|U|$ of a hypergraph and the lower bound is 1 vertex

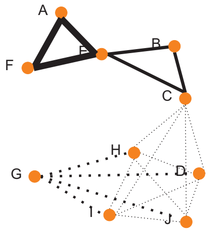

Figure 1. Graph representation of group of users. Different level of trust between users is shown in a different line format.

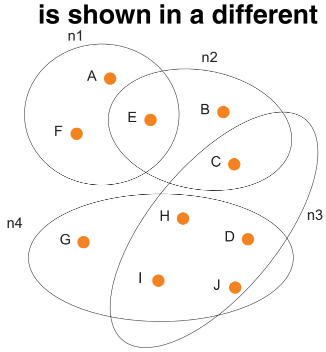

Figure 2. Hypergraph representation of group of users. Different level of trust is shown by ovals corresponding to the groups of users (hyperedges).

Figure 3.1. As the level of trust differs between users G,H,I,J,D (shown in a bold dashed line) and between users H,I,J,D,C the hypergraph in Figure 3.1 contains two distinct hyperedges $n3$ and $n4$ with different level of trust. Moreover, in graph model (Figure 3.1) user H has to store and maintain 5 relationships (corresponds to the size of its adjacency set). In hyperaph model (Figure 3.1) this is reduced to 2 – number of groups user H is member.

3.2 G2H Algorithm

As the security model maintains dynamic DVOs as hypergraphs, it is necessary to propose a transformation of a general input structure into hypergraphs.

The transformation cannot be done arbitrary but it must consider a semantics of the input graph – social relationships among users. The main task of the transformation is to identify highly correlated substructures and transforms them into a groups of users (hyperedges).

For details readers are referred to our foregoing papers [18],[22]

3.3 SD Algorithm

The dynamics of the model corresponds to the fact that users[3] must react to changes they are posed to in their real lives. Therefore, this section introduces the dynamic part of the security model – the *SD algorithm*.

The input of the SD algorithm is a quadripple (u_1, n_1, u_2, n_2) where the following holds: user u_1 from group n_1 is invited by user u_2 from group n_2 to group n_2.

Algorithm 1 The SD algorithm

1: **procedure** RUNSD(u_1,n_1,u_2,n_2)
2: **if** ($n_1 \cap n_2 = \oslash$ or $W_{n_1} = (\Lambda \pm W_{n_2})$) **then**
3: add(u_1 into n_2)
4: **end if**
5: **if** ($W_{n_1} < \Lambda \pm W_{n_2}$) **then**
6: n_{NEW}=SplitNet(u_1,u_2,n_2,$n_2 \cup n_1$)
7: **end if**
8: **if** ($W_{n_1} > \Lambda \pm W_{n_2}$) **then**
9: n_{NEW}=SplitNet(u_1,u_2,n_1,$n_2 \cup n_1$)
10: **end if**
11: MergeNets(n_1,n_2,N_{NEW})
12: **end procedure**
13: **procedure** SPLITNET(u_1,u_2,n_x,Φ)
14: $n_{x_{OLD}} = u_1, u_2 \cup (n_x - \Phi)$
15: $n_{x_{NEW}} = n_x - \{u_1, u_2\}$
16: $W_{n_{NEW}} = W_{n_x} ++$
17: return N_{NEW}
18: **end procedure**
19: **procedure** MERGENET(n_1,n_2,n_{NEW})
20: **for all** $n_i, n_j \in \{n_1, n_2, n_{NEW}\}$ **do**
21: **if** ($|n_i \cap n_j| \pm \epsilon > min(|n_i|, |n_j|)$
22: **and** $W_{n_i} = \Lambda \pm W_{n_j}$) **then**
23: $n_i = (n_i \cup n_j)$
24: $W_{n_i} --$
25: **end if**
26: **end for**
27: **end procedure**

In Algorithm 1 is the SD algorithm described in a pseudo-code. The input is a new invitation issued by u_2 from group n_2 for u_1 from n_1. The *RUNSD procedure* firstly identifies whether exists an intersection between the groups. The fact that intersection does not exist or the groups have the same level of trust (line 2), implies that user u_1 can be added with no harm.

If an intersection exists or groups differ in the level of trust, the procedure preserves the local security by splitting the groups (line 6 or 9).

Consequently, *MERGENETS procedure* checks the sizes of the intersections between the groups involved. If size of

[3]Term user will be extended in our case so that it will stand for human users as well as machines or computer agents, ...

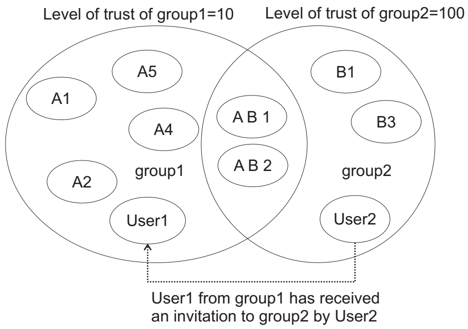

Figure 3. Initial configuration

the intersection is larger than a threshold ϵ the groups are merged. Parameters Λ and ϵ drive the algorithm causing more splitting or more merging (see section 4).

Figures 3.3 and 3.3 graphically show an example scenario. At the beginning (Figure 3.3) there are two groups with different level of trust and two users in the intersection ($AB1, AB2$). In Figure 3.3 is shown the final state. Whereas $group1$ remains unchanged, group $group2$ is divided into *group2 old* and *group2 new*. Such splitting corresponds to the fact that users $AB1$ and $AB2$ do not accept addition of $User1$ from untrusted group $group1$ into trusted $group2$. Moreover, $User2$ who issued the invitation become also suspicious. On the other hand, the other members of $group2$ become members of both groups (*group2 new* and *group2 old*) creating a bridge for further identification or separation of groups members.

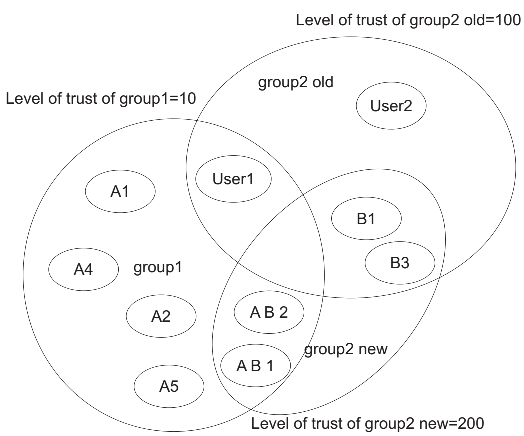

Figure 4. Situation after splitting

4 Experimental Results

The main purpose of the experiments is to verify that the SD algorithm preserves long-time stability and is able to stabilize structure of DVO at some point in time.

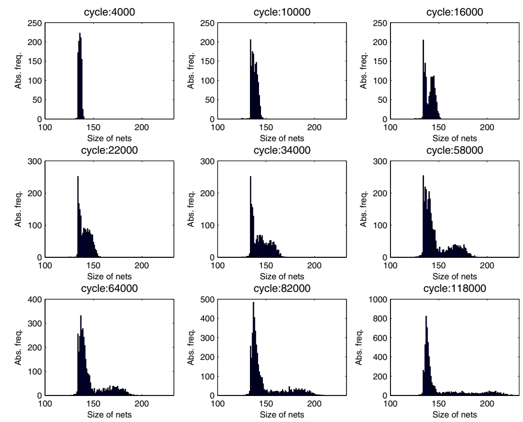

Figure 5. Histograms for Λ=1, ϵ=1, starting amount of groups=908

That stability of the model directly influences its usability is rather clear. For instance, assume that a proposal is not stable so that it tends to create one huge group containing all users. In such situation all users have access to data of all others. In the opposite situation of many very small groups, there are few users that might seamlessly share information (members of the same group).

The SD algorithm can be driven by two parameters:

- Λ - influences the merging procedure. Higher Λ implies lower probability of merging.
- ϵ - controls the splitting procedure. The higher ϵ implies the higher probability of splitting.

The input to the SD algorithm was created from records of calls held in a real mobile network in the Slovak republic. The records were quartets (*recipient*, *sender*, *type of the request*, *duration*). For our experiments we extracted pairs (*recipient*, *sender*), which mean in our interpretation: the sender (u_2) invites the recipient (u_1) to one of its groups (chosen randomly). For the experiment 161 404 records of phone calls among 121 672 users were extracted, which equals to 161 404 dynamic changes in the system of groups.

In the first three experiments, the initial system configuration consists of 908 groups each containing 134 users with the equal level of trust. In the last case the initial configuration consists of 227 groups.

The Figure 3.3 shows the evolution of the system of the groups in histograms where on axis-y is shown absolute frequency and on axis-x are shown sizes of groups (note that we extracted only important or interesting situations). At the beginning one can see sudden changes in the shape of the histograms. Nevertheless, round cycle 90 000 systems achieves a stable configuration and the remaining histograms contain minor changes. The Figure 4 shows the

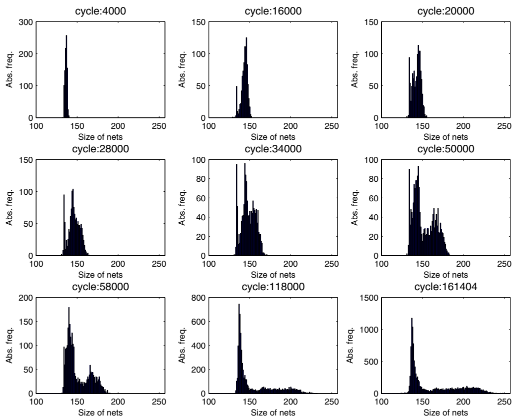

Figure 6. Histograms for Λ=1, ϵ=3, starting amount of groups=908

evolution for different parameters, particularly, Λ=1, ϵ=3. The stability is achieved a bit latter round cycle 130 000.

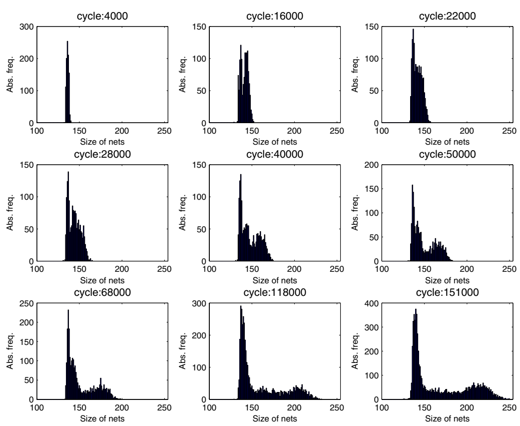

Figure 7. Histograms for Λ=3, ϵ=1, starting amount of groups=908

The penultimate combination of the parameters were Λ=3, ϵ=1. This combination prefers splitting to merging. The stability of achieved also round cycle 130 000, but the absolute frequencies are rather lower compared to the previous combinations, while bigger Λ prefers splitting to merging.

The last combination of the parameters was $\Lambda = 1$ and $\epsilon = 30$. With these parameters the SD algorithm tends to process more merging and little splitting. The histograms given in Figure 4 shows the expected behavior since the merging part of the SD algorithm tends to create one huge group.

The experiments showed that in case of low differences in parameters, the SD algorithm tends to achieve stability

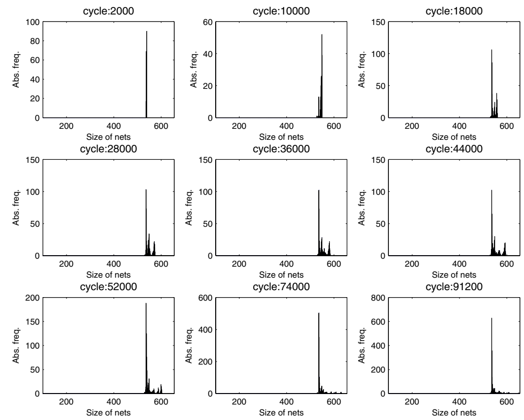

Figure 8. Histograms for Λ=1, ϵ=30, starting amount of groups=227

mostly round cycle 130 000. The main differences between various parameters are mainly visible in the first half of the histograms. After the first half (round cycle 70 000) the shapes do not differ much and the system remains stable for the rest of the experiment and the main differences are in absolute frequencies.

5 Conclusion

The paper presents an approach for treating trust in a distributed and dynamic environment of the Internet. The approach takes advantages of the reputation systems based on social networks together with the advantages of weighted hypergraphs for storage and management of groups of users organized in dynamic Virtual Organizations. The model is naturally distributed. The most important question whether the model can be consistently developed was positively answered by our experiments with a real data as the inputs.

Acknowledgment

The work was supported by the project 1ET100300419 of the Program Information Society (of the Thematic Program II of the National Research Program of the Czech Republic) “Intelligent Models, Algorithms, Methods and Tools for the Semantic Web Realization”, partly by the Institutional Research Plan AV0Z10300504 “Computer Science for the Information Society: Models, Algorithms, Applications” and by Ministry of Education, Youth and Sports of the Czech Republic, project No. 1M0554 Advanced Remedial Technology and Processes.

References

[1] J. Basney, W. Nejdl, D. Olmedilla, V. Welch, and M. Winslett. Negotiating trust on the grid. *In 2nd WWW Workshop on Semantics in P2P and Grid Computing,New York, USA*, may 2004.

[2] M. Y. Becker and P. Sewell. Cassandra: distributed access control policies with tunable expressiveness. *In 5th IEEE International Workshop on Policies for Distributed Systems and Networks, Yorktown Heights*, June 2004.

[3] P. Bonatti and P. Samarati. Regulating service access and information release on the web. *In CCS 00: Proceedings of the 7th ACM conference on computer and communications security, ACM Press*, page 134143, 2000.

[4] P. A. Bonatti and D. Olmedilla. Driving and monitoring provisional trust negotiation with metapolicies. *In 6th IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY 2005), Stockholm, Sweden, IEEE Computer Society*, pages 14–23, jun 2005.

[5] E. Damiani, S. D. C. di Vimercati, S. Paraboschi, P. Samarati, and F. Violante. A reputation-based approach for choosing reliable resources in peer-to-peer networks. *In Proceedings of ACM Conference on Computer and Communications Security*, page 202216, 2002.

[6] C. Duma, N. Shahmehri, and G. Caronni. Dynamic trust metrics for peer-topeer systems. *In Proceedings of 2nd IEEE Workshop on P2P Data Management, Security and Trust (in connection with DEXA05)*, August 2005.

[7] C. M. Ellison, B. Frantz, B. Lampson, R. Rivest, B. M. Thomas, and T. Ylonen. Spki certificate theory. *Internet RFC 2693*, October 1999.

[8] S. Garfinkel. Pgp: Pretty good privacy. *O'Reilly & Associates, Inc.*, 1995.

[9] M. Golumbic. *Algorithmic graph theory and perfect graphs*. Academic Press, 1980.

[10] T. Grandison and M. Sloman. Survey of trust in internet applications. *IEEE Communications Surveys*, 3(4), 2000.

[11] M. Gupta, P. Judge, and et al. A reputation system for peer-to-peer networks. *Thirteenth ACM International Workshop on Network and Operating Systems Support for Digital Audio and Video, Monterey, California.*, 2003.

[12] L. Kagal, T. Finin, and A. Joshi. A policy based approach to security for the semantic web. *In Proceedings of the 2nd International Semantic Web Conference, Sanibel Island, Florida,USA*, Oct. 2003.

[13] S. Kamvar, M. Schlosser, and et al. The eigentrust algorithm for reputation management in p2p networks. *WWW, Budapest, Hungary*, 2003.

[14] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina. Eigenrep: Reputation management in p2p networks. *In Proceedings of 12th International WWW Conference*, page 640651, 2003.

[15] S. Lee, R. Sherwood, and et al. Cooperative peer groups in nice. *IEEE Infocom, San Francisco, USA*, 2003.

[16] N. Li and J. Mitchell. A role-based trust-management framework. *In DARPA Information Survivability Conference and Exposition (DISCEX), Washington, D.C.*, Apr. 2003.

[17] J. Pujol, R. Sanguesa, and et al. Extracting reputation in multi agent systems by means of social network topology. *First International Joint Conference on Autonomous Agents and Multi-Agent Systems, Bologna, Italy.*, 2002.

[18] M. T. R. Špánek. Secure grid-based computing with social-network based trust management in the semantic web. *Neural Network World, International Journal on Neural and Mass-Parallel Computing and Information Systems*, 16(6):475–488, December 2006.

[19] J. Sabater and C. Sierra. Regret: A reputation model for gregarious societies. *4th Workshop on Deception, Fraud and Trust in Agetn Societies, Montreal, Canada.*, 2001.

[20] Steiner, C. . Neuman, and J. . Schiller. Kerberos : An authentication service for open network systems. *in Proc. Winter USENIX Conference, Dallas*, 1988.

[21] G. Tonti, J. M. Bradshaw, R. Jeffers, R. Montanari, N. Suri, and A. Uszok. Semantic web languages for policy representation and reasoning: A comparison of kaos, rei and ponder. *In Proceedings of the 2nd International Semantic Web Conference, Sanibel Island, Florida,USA*, Oct. 2003.

[22] R. Špánek and M. Tůma. Secure grid-based computing with social-network based trust management in the (semantic)web. *in Proceedings of The 8th International Conference on information Integration and Web-based Application & Services (iiWAS2006)*, 2006.