A Service-Oriented Fuzzy Reputation System to Increase the Security of a Broadband Wireless Metropolitan Network

Alexandre Gomes Lages, Flávia C. Delicato, Gizelle Kupac Vianna, and Luci Pirmez
NCE – Federal University of Brazil (UFRJ)
P. O Box 2324, Rio de Janeiro RJ, 20001-970, Brazil
http://www.labnet.nce.ufrj.br
{alexandrelages, fdelicato, gizellekupac, luci} @nce.ufrj.br

Abstract. The increasing development of the Wireless MAN technologies will allow the exchange of services directly among subscriber stations. Schemes to assure the safety of services should preferably adopt a distributed architecture, such as the Reputation Systems architectures, commonly used in Peer-to-Peer, in order to keep the network scalability. Reputation Systems have the goal of evaluating individual peers, relying on their previous interactions in the network. We propose the use of a service oriented Reputation System based on Fuzzy Logic, as a solution for increasing the safety level of a Wireless MAN. The paper presents the description of the proposed system and the simulations performed to evaluate it in a wireless network environment.

I – INTRODUCTION

The IEEE standard 802.16 defines the MAC protocol and air interface specification for wireless metropolitan area networks (MANs), aiming to meet the demand requirements of high quality multimedia services, associated with a high availability. The standard promises to expand the market of broadband connections, through a significant decrease of needs in infrastructure investments, an immediate access distribution, combined with lower service prices to end users.

The mechanisms available in wireless MANs to provide security are defined in the Privacy Sublayer. Such sublayer provides authentication, secure key exchange and encryption mechanisms, using the MAC layer, to guarantee the safety of exchanging messages among stations. However, besides providing security at the access level, primitives are required in order to increase the security on the service level during the interactions among stations. Therefore, additional mechanisms are needed to increase the security. The use of a distributed approach, such as the Reputation Systems (RS) [1], to increase the security of a peer in a Peer-to-Peer (P2P) Network is an afforded choice against the mechanism of Public Keys, which has a centralized solution and a limited scalability. RSs [2, 3, 4] provide a way for building trust relying on peers past experiences, as a reputation value is returned after a peer uses a service. The main function of a RS is to allow for a peer the judgment about other peer, based on the quality and the reliability of transactions they performed before. A given station could use many services from many proveder stations, with probably different reputation values related to each provider, and these values are stored on several stations over the network. Therefore, mechanisms to compute the final reputation of a given station as well as the use of a fast protocol to retrieve all the reputation values belonging to a single station become necessary. The avoidance of distributing the reputation value in many stations can minimize the costs of this search. A possible solution is to define, for each station, a sub-set of stations responsible for storing its reputation, thus constraining the search to such sub-set. Unfortunally, since a reputation is created based on peers' opinions, a group of them could increase or decrease reputation of other peers in the network, in a formation called **Collusion**. Another problem with RSs is their dynamic nature. Peers can enter and leave the network dynamically, and unless an infrastructure is provided for keeping redundant information about peers' reputation, the values stored in these peers will eventually be lost. Despites these facts, a RS is an efficient choice to provide security for services exchange [2, 3, 4].

Until the present moment, most RSs assign reputation values to isolated peers instead of adopting a service-oriented approach that sets a reputation value to a specific tupla *peer-service*. We propose the use of a RS using a service-oriented approach, in a Wireless MAN with Mesh topology, as a mechanism to increase the network security. The following topics will be covered: (i) the message exchange mechanism used to discover a *peer-service* reputation; (ii) the computation of the reputation itself; (iii) the process used to avoid the collusion; and (iv) results of simulation performed with the prototype. This work brings in two main contributions: the mechanism used to compute the *peer-service* reputation, and the process of updating a peer reputation. Differently from previous approaches [2, 3], the update process will be done based on both past and new reputation values.

The remainder of this work is organized as follows: in Section II we introduce our mechanism to increase the security of a Broadband Wireless

Metropolitan Network. In Section III, the reputation calculation using Fuzzy Logic is presented. In Section IV we provide simulation results, and Section V we make some concluding remarks.

II – Safety Model for Broadband Wireless Metropolitan Networks

One important goal of our work is to propose a reputation-based security system for *Mesh* metropolitan wireless networks. Our service-based approach has an important role in providing safety for these networks, as it supports the secure exchange of services directly among subscriber stations. Mesh Wireless Broadband Networks are characterized by allowing the direct interaction between stations, with no need for a centralized infrastructure. A paramount requirement for this kind of interaction is to guarantee the safety of communicating stations. IEEE 802.16 standard addresses the security at the link layer level. However, to fully explore the potential and flexibility offered by a Mesh network, it is important to include mechanisms to provide security at the application or service level. Reputation Systems (RSs) are a promising approach to provide such level of security.

In general, RSs use the peers' previous experience to assign reputation levels to resources or other *peers*. Such systems assume that peers maintain the same identifier during all the time they remain inside the system. In several works [2, 3, 4], the reputation value is assigned by the user, after an interaction with a peer. By adopting a service-oriented approach, the reputation value is a function that represents the use of a service by a peer. For keeping the integrity of the information stored by them, each *peer* is assigned a a single public key. For that purpose, we adopted the model SPKI/SDSI [8, 9] which has a decentralized approach for the process of keys authentication. For storing the information about interactions with other subscribers we defined the *Service Reputation Degree-Table (SRDT)*, located at each peer. This table has four fields: the service class identification (IDS), which can be *UGS*, *rtPS*, *nrtPS*, or *BE*; the peer identification (IDP); the peers' Reputation degree in relation to this service class (REP); and the Relationship degree (REL) among the IDP and the peer storing the table. The fields REP and REL represent previous interaction experiences with the service under consideration, and they assume values in the interval [0,1]. Whenever a peer uses a service, REP is updated based on the performed actions.

As time elapses, the *peer* that hosts the service (HP) updates the fields REL of the SRDT, in order to indicate its level of interaction in relation to each *peer*. The closer the value of REL is to one, the greater would have been the interaction of this specific service with the *peer*, and the more trustable the corresponding value stored in the field REP will be. The value of this field can be either incremented or decremented, depending on the interactions carried out among the *peers*. When a *peer* needs to use another *peers'* service for the first time, two situations can occur: the *peer* has an invitation or not. In the first situation, the *peer* uses the service invitation sent by another *peer* that already uses this service, and the new peer inherits the reputation of the inviter. A service invitation is represented by a message, containing the public key of the inviter *peer* and its IDS. When the HP receives a request by a *peer* having an invitation, the HP will first check its authenticity with the inviter, in a process out of the scope of this work. In the second situation, the HP has to take the decision of providing or not the service to the new *peer*, using specific policies as, per example, quering the *peers* belonging to their relationship network and checking whether any of them has informations about the pair *peer-service*.The possibility for a new *peer* to access a service, even without having an invitation, makes it possible for new *peers* to join the network without any associated reputation value. After using the service, the behavior of the requesting *peer* (RP) is evaluated, and the result is reflected on the REP field of the HP's table SRDT.

One class of services comprises several applications but, for the sake of simplicity, we assume that each service class comprises only one application. In addition, the Aggregate reputation value is calculated by first calculating the *peer's* reputation in each service class. Different weights, representing priorities, are associated to each service class. For example, the weight of class UGS can be 4 and the one of class BE, 1. In Section 5, the calculation of one peer's reputation is described.

To use a Reputation System in a network with Mesh topology, it is necessary the use of a mechanism to exchange messages about the reputation that are been calculated and stored within the stations. In this work, a structured P2P overlay network was used, offering the primitives that make possible the exchange of reputation values between stations. There are several architectures that offer a structured network [5, 6, 7]. Although a structured network incur high maintenance cost of the peers, it was used because an unstructured network has a high lookup cost and use a broadcast mechanism to find other peers, with the impact in the bandwidth used.

We adopted *Chord* protocol [5] for searching the reputation value of a given *peer-service* pair. *Chord* is a scalable protocol that supplies message search mechanisms in P2P networks, using a ring topology, and *keys* generated through a *hash* algorithm. When a *peer* wishes to use another *peer's* service, the HP applies one or more *hash* algorithms to find out the keys that identify the *peers* storing the reputation

(SP's) of the RP. The *peer* which has the same key is responsible for the information storage. To increase the redundancy in the storage, two or more *hash* functions can be used, so the information can be stored in two or more *peers*. Figure 1 presents the network structure used in this article. The figure shows peer S requesting a service of *peer* A (continuous line). Upon receiving the request, *peer* A executes two *hash* algorithms in order to obtain two distinct keys, and afterwards, it sends the requisitions for these *peers* asking about *peer* S reputation (traced lines).

A differential in our proposal consists in the adoption of an additional step, besides those executed by *Chord* protocol, aiming to increase the reliability of reputation information. In this additional step, the HP (in the example *peer* A) sends requests to the *peers* considered "Friends" (*peer* C), i. e., *peers* that have a high value in the field REL, as an attempt to improve the reputation evaluating (1) of the RP (*peer* S); and (2) of the SP's of the RP (*peers* B and N). Considering Figure 1, C will return the reputation of S, in case it has it, and also the reputation of *peers* B and N, the SP's of S. The requests for *peers* considered "Friends" is justifiable, since a "Friend" can have a more reliable value of REP for the RP than other *peers* returned by the *hash* algorithm. The REP returned by these "Friends" also informs the REP of the *peers* returned by the *hash* algorithm, in case the HP does not have it. If is neither possible to obtain the REP values from the SP, nor from "Friend" *peers*, the SP will have a low value, next to zero, for the field REL, so that the influence of the REP returned by them will be considered very low in the calculation of final reputation of the RP.

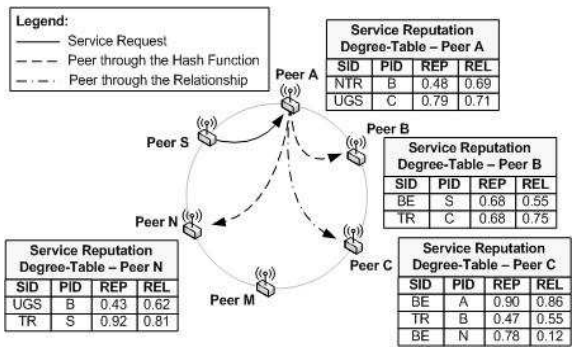

Figure 1. Communication Module and Reputation Calculation

III – Reputation Calculation using Fuzzy Logic

This section will present two inference processes. One process is related to the computation of a *peer* reputation, and the other one is used to avoid the formation of collusion in the network.

The fields REP and REL of the SRDT are used for calculating a *peer's* reputation. A *peer*, upon receiving a service request, sends requisitions about the RP's reputation to the SP's, and also to the *peers* considered its "Friends". The returned REP values always refer to the requested *peer-service* pair, and the HP also checks its SRDT values for REL those SP's. Upon acquiring all this information, the HP can perform the calculation of the RP's reputation. When the REL value is low, for example, because it refers to an unknown *peer*, the REP value returned by this *peer* will have a low influence on the reputation calculation of the RP.

The procedure for the calculation of a peer's reputation is based on the Fuzzy Logic and is called "Fuzzy Reputation Calculation". By definition, a fuzzy variable is defined by the quadruplet (X, R, U, M), where: X is the the variable name; R is the set of linguistic values of X; U is the discursive universe of variable X; M is a semantic rule that associates each linguistic value to its M(r) meaning. The first step of the calculation comprises the definition of fuzzy variables and their correspondind fuzzy sets. After refining the initial definitions, through many simulations, the following variables were selected:

Reputation-Degree (input variable): contains a *peer's* reputation degree returned by a *peer* that has provided the service ***(Very High, High, Medium, Low, Very Low)***.

Relationship-Degree (input variable): contains the relationship degree between the *peer* that has sent the reputation and the *peer* in charge of storing it ***(Friend, Colleague, Stranger)***.

Final-Reputation (output variable): represents the *peer's* reputation in relation to the relationship level between them ***(Very high, High, Medium, Low, Very Low)***.

The combination of all fuzzy input variables can generate up to 15 possible IF-THEN rules in the diffuse rule base. The surface (a) of Figure 4 graphically synthesizes the adopted rules and presents the returned reputation value. For evaluating the rules, we used a *Mamdani* system type and the following methods: (i) *And* - min; and (ii) *Or* - max; (iii) *Implication* - min and (iv) *Aggregation* - max. The chosen *Defuzzification* method was the *mom* (average of the maximums), because it presents the best results and coherence with the output. The diffuse rule machine determines which rules will be activated by the fuzzy input variables in order to determine the output fuzzy sets that will be defuzzified.

The result of the "Fuzzy Reputation Calculation" procedure, the Fuzzy variable$_{i,j}$, is then used by the equation of Figure 2 for calculating the *peer's* local reputation, i. e., the reputation of the *peer* in a determined service class. In the equation, $\boldsymbol{Reputation_j}$ represents the local REP of *peer* j; $\boldsymbol{Relationship_{i,j}}$ represents the REL value among *peer* i and *peer* j. The $\boldsymbol{Fuzzy\ variable_{i,j}}$ represents the value calculated by *peer* i's inference machine in relation to j's. Since a *peer* can receive several

reputation values about another *peer* from the SP's and from the friend *peers*, for the determination of the *peer's* final reputation, a weighted average of the REL levels of the *peers* that returned the final REP is performed.

$$\text{Reputation}_j = \frac{\sum_{i=1}^{n} \text{Relationship}_{i,j} * Fuzzy_{i,j}}{\sum_{i=1}^{n} \text{Relationship}_{i,j}}$$

Figure 2. Formula for the calculation of a peer's reputation

For example, in Figure 1 *peer* S requests a service from *peer* A. *Peer* A, when executing the *hash* algorithm, returns *peers* N and B as SP of S. The reputation values of S returned by N and B are 0.92 (Very High) and 0.68 (High), respectively. *Peer* A knows only *peer* B, and its REL with B is 0.69 (Friend). *Peer* C returns the information that *peer* N is in its table, and its REP is 0.78 while its relationship value is 0.12 (Unknown). The REP and REL information of B stored in C are also sent to *peer* A (0.47 and 0.55, respectively). Next, the inference machine is used to calculate *peer* A's reputation in relation to B ($Fuzzy_{a,b}$) and to N ($Fuzzy_{a,n}$). For A in relation to B, the "Fuzzy Reputation Calculation" procedure receives as scalar input variables the values 0.68 and 0.69, for REP and REL, respectively. One of the rules that can be applied to these entries is: **If** (A's reputation in relation to B's) is **HIGH** and (A's relationship in relation to B) is **FRIEND then** Final-Reputation is **HIGH**. The defuzzification process returns, for the output fuzzy variable **reputation**, the scalar 0.59.

For calculating A's reputation in relation to N, the fuzzy procedure receives as scalar input variables the respective values, 0.92 and 0.12, for the variables REP and REL. The rule that can be applied is: **If** (A's reputation in relation to N) is **VERY HIGH** and (A's relationship in relation to N) is **STRANGER then** Final-reputation is **MEDIUM**. The defuzzification process returns the REP value of 0.56. Once having these values, *peer* A can calculate the final reputation of S applying the formula in Figure 2: S = (0.69*0.59)/(0.69+0.12) + (0.12*0.56)/ (0.69+0.12) = 0.58.

In order to perform the calculation of an ***Aggregate*** *peer* reputation (AREP), all REP values a *peer* has in the four service classes are aggregated to generate one single reputation value. As we have already mentioned, service classes have different priorities. Thus, a *peer* having a high reputation in a low priority class and a low value in a high priority class can have as a result a low value of AREP. To connect these confronting values, we used a weighted average of REP values. Due to restrictions in space, the discussion about the launching of the evaluation process of a *peer*, after the utilization of a service, is outside the scope of this article.

Received REP values that are outside a certain percentage from the distribution of values at SRDT are discharged and do not result in table updating, in order to avoid a well known peer from being penalized by another one that informs a low REP value for it, outside its historic average and standard deviation. However, such values are inserted in a base of historic REP values per service and per *peer* (HREP) received from a given *peer*. This base is used to allow for successive REP values outside the standard the possibility to change the distribution pattern, in a future moment. In this article, the HREP values are modeled using a normal distribution.

When a *peer* receives a REP value in accordance with the distribution pattern, before storing this value into the SRDT, a second process is carried out to avoid oscillations in REP, and so decreasing the chance of collusion. Thus, when a *peer* receives messages for storing REP values, the REL value of those *peers* sending the messages has also to be taken into consideration, by using the expression in Figure 3.

In this formula, $\mathbf{Rep_{average}}$ is the *peer's* average REP value, $\mathbf{Rep_{new}}$ is the new received REP value, and the result is attributed to $\mathbf{Rep_{peer}}$. Variable α can receive values in the interval [0,1]. If α is close to 1, the $Rep_{average}$ will be more relevant, and the collusion is avoided. Otherwise, Rep_{new} will have a greater weight.

$$Rep_{peer} = \alpha * Rep_{average} + (1 - \alpha) * Rep_{new}$$

Figure 3. Formula to avoid oscillations in the update of reputation degree

An important differential of our work is related to the proposed method to obtain the value of the variable α. To establish such value, it must be taken into account the REL value among the peer sending the REP of the RP and the HP. So, in case the HP receives a message from a *peer*, and the REL degree between them is considered High ("Friend"), the value of α should be low for the value Rep_{new} to have a higher weight in REP updating. If the *peers* sending the reputation values have a low REL with the SP, α will be high, and the updating impact will be low. The surface of Figure 4 (B) presents the generated values of α, upon the REL and REP values received by a *peer*. This surface was generated by a fuzzy inference machine with Degree-Reputation (linguistic variables: Very High, High, Medium, Low and Very Low) and Relationship-Degree (linguistic variables: Friend, Colleague and Stranger) variables, and Alfa output variables (linguistic variables: Small, Medium and High)

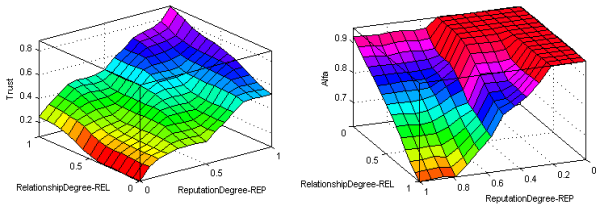

Figure 4. Surface for the calculation of Trust (A) and Alfa (B)

IV – System Evaluation

We performed a set of simulations in order to analyse the capability of the proposed mechanism to avoid the collusion. Simulations were conducted by using the Network Simulator (NS-2) [10] and a wireless network 802.11 with a total of 60 peers in a range of 100mx100m. Since in the performed simulations we were interested in measuring the reputation of a *peer* at the application leve, it was not necessary to use the Chord Protocol and an 802.16 network infrastructure. To analyze the benefits of the proposed mechanism, the tests compared the process of updating the SRDT using a fixed α, as in previous works [2, 3], that also does not discard values out of the distribution, with the proposed *variable*-α method.

Figure 5 illustrates the variation of REP values a *peer* A received to store, related to peer B. We used only one hash function, which always returns *peer* A. Simulations were accomplished with collusion sizes of 5%, 10%, 20%, 30%, 40%, and 50% of network. The initial REP of peer B was set to 0.5. Thirty rounds for each size of collusion were made and the average, standard deviation and confidence interval of 95% calculated. The total simulation time was 3000 seconds, with 300 seconds for the transient phase. For each collusion group, a set of random *peers* was chosen, from a total of 60 *peers*, and it was selected from this set: (i) a RP (*peer* B); and (ii) the *peers* that create the collusion. At each 5 seconds, peer B generates an event to access the service of another *peer* in the network, in a random way. During the first 300 seconds of simulation, the collusion group did not affect the REP of *peer* B. After that time, when *peer* B access the service from a *peer* that belongs to the collusion group, a bad value of REP was returned, between Very Low and Low, independently of actions that *peer* B has made. If the *peer* did not belong to the collusion, it would return the real REP of peer B. Figure 5 (a) illustrates the final average REP of *peer* B, using the equation of Figure 3 (with variable α) and using the values 50%, 75%, 85% and 95% as a parameter to the distribution. Figure 5 (a) also depicts the average initial REP of *peer* B, before the creation of collusion. Comparing the curves in the Figure 5 (a), it is evident that even though with a collusion size of 20%, the final REP of *peer* B did not have a negative difference compared with its initial REP. Only after collusion size increased to 25% the value of final REP of *peer* B became negative in relation with the initial and final value. The curve using the parameter 50% declines more quickly than the others because it adopts a more conservative strategy. With the increasing number of *peers* in collusion, the difference between REP values returned became large and more values of REP are discarded. According to Figure 6 (a), when using the parameter of 50%, the percentage of discart is approximately 73%, with a collusion group of size 50% of network. It confirms why the final REP of *peer* B in Figure 5 (a) decreases so fast.

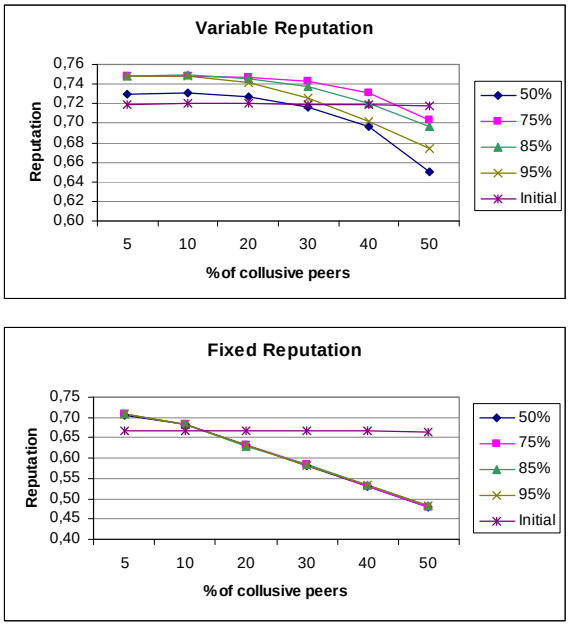

Figure 5. Updating the Table Service Reputation – (a) Fuzzy (b) Fixed α

Figure 5 (b) shows the final average REP of *peer* B, using a fixed α value of 95%. The four curves are overlapped because there is not a mechanism to discard REP values. This figure also shows the initial average REP of *peer* B, before the formation of collusion. Comparing the four curves of final REP with the curve representing the initial REP, the final reputation begins to present a difference negative, after a collusion size of approximately 12%. The difference between initial and final REP is very large when compared with *variable-α-and-discard* mechanisms. Figure 6 (b) compares the initial and final REP using the *variable-α-and-discard* mechanism with the *fixed*-α method. With a collusion group of 30% of the network, the difference between initial and final REP using the proposed schema is almost zero. When using *fixed*-α, the difference is around 12.5% negative. For a collusion group size around 50%, the values are approximately 9% and 28%, with a best result using the *variable-α-and-discard* mechanism, respectively. Comparing the two methods, when the collusion size increases, the final REP of *peer* B using *fixed*-α presents a linear decreasing in REP. The REP

became negative with a collusion size of 11% using *fixed*-α, while the same happened, for a distribution value of 75%, only when the collusion size is around 30%. Once using the *variable*-α-*and*-*discard* mechanism, the curve decreases more smoothly, with the final REP decreasing around 9% for a collusion size of 50%. Summing up, the proposed process for updating SRDT is more resilient than other approaches.

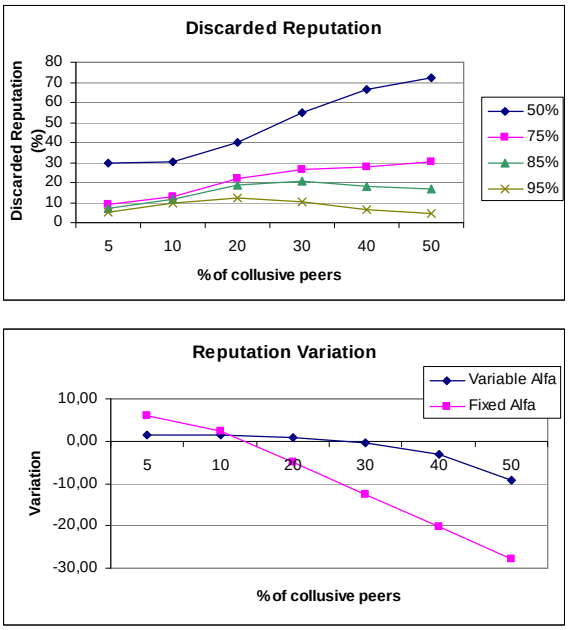

Figure 6. (a) Discarded Reputation Values (b) Variation of Reputation Values

V – Concluding Remarks

We have presented a new approach for increasing the security at the service level of broadband wireless network. The following features and strategies have been selected to support our goals: (i) the utilization of a RS to manage the access control of services offered in the network; (ii) the adoption of a service-oriented approach to define levels of trustworthy, not only to isolated peers but to the pair *peer*-*service*; and (iii) the use of fuzzy logic to compute the reputation assigned to a *peer*-*service*. The mechanism of message exchange related to a reputation of a *peer*-*service* was supported by a well known protocol, however augmented with additional steps. In this way, stored reputation values in *peers* belonging to the same network are shared and used. The reputation values of each *peer*-*service* are dynamically changed, according to transactions performed by the *peer*. The updating of these values is based on mechanisms that avoid or minimize the collusion, and, at the same time, decrease the oscillation of the reputation values stored. We reported initial simulation-based experiments, demonstrating the feasibility, effectiveness, and benefits of our approach.

VI – References

[1] Resnick, P., et al. (2000) "RS's". In: Communications of the ACM, Vol. 43, December.

[2] Song, S., Hwang, k., Kwok, Y. and Zhou, R. (2005) "Trusted P2P Transactions with Fuzzy Reputation Aggregation". In: Security in P2P Systems, IEEE Internet Computing, November-December.

[3] Kamvar, S., Schlosse, M. and Garcia-Molina, H. (2003) "The EigenTrust Algorithm for Reputation Management in P2P Networks". In Proceedings of the Twelfth International World Wide Web Conference, May.

[4] Despotovic, Z. and Aberer, K. (2005) "P2P Reputation Management: Probabilistic Estimation vs. Social Networks". In: Journal of Computer Networks, Special issue on Management in Peer-to-Peer Systems: Trust, Reputation and Security, Elsevier.

[5] Stoicay, I., et al. (2003) "Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications". In: IEEE/ACM Transactions on Networking, ACM Press, vol. 11, no. 1, pp. 17–32.

[6] Rowstron, A. and Druschel, P. (2001) "Pastry: Scalable, distributed object location and routing for large-scale *peer*-to-*peer* systems". In: IFIP/ACM Int. Conference on Distributed Systems Platforms, Germany, pp. 329-350, Nov.

[7] Zhao, B., et al. (2004) "Tapestry: A Resilient Global-Scale Overlay for Service Deployment". In: IEEE Journal on Selected Areas in Communications, Vol 22, No. 1, January.

[8] Ellison, C. M., Frantz, B., Lampson, B., Rivest, R., Thomas, B. M., and Ylonen, T.(1999). "SPKI Certificate Theory". Internet Engineering Task Force RFC 2693.

[9] Rivest, R. L. and Lampson, B. (1996). "SDSI – A simple distributed security infrastructure".Presented at CRYPTO'96 Rumpsession.

[10] Network Simulator (2005), http://www.isi.edu/nsnam/ns/, December.