Trust Evaluation in Ad-Hoc Networks

George Theodorakopoulos
gtheodor@isr.umd.edu

John S. Baras
baras@isr.umd.edu

Electrical and Computer Engineering Department
and the Institute for Systems Research
University of Maryland
College Park, MD 20742

ABSTRACT

An important concept in network security is trust, interpreted as a relation among entities that participate in various protocols. Trust relations are based on evidence related to the previous interactions of entities within a protocol. In this work, we are focusing on the evaluation process of trust evidence in Ad Hoc Networks. Because of the dynamic nature of Ad Hoc Networks, trust evidence may be uncertain and incomplete. Also, no pre-established infrastructure can be assumed. The process is formulated as a path problem on a directed graph, where nodes represent entities, and edges represent trust relations. Using the theory of semirings, we show how two nodes can establish an indirect trust relation without previous direct interaction. The results are robust in the presence of attackers. We give intuitive requirements for any trust evaluation algorithm. The performance of the scheme is evaluated on three topologies.

Categories and Subject Descriptors

C.2.0 [**Computers-Communication Networks**]: General—*Security and Protection*; G.2.2 [**Discrete Mathematics**]: Graph Theory—*Path and circuit problems*

General Terms

Design, Security

Keywords

ad-hoc networks, semirings, trust evaluation, trust metric

1. INTRODUCTION

The notion of trust, in the realm of network security, will for our purposes correspond to a set of relations among entities that participate in various protocols [11]. Trust influences decisions like access control, choice of public keys, etc. Trust relations are determined by rules that evaluate, in a meaningful way, the evidence generated by the previous behavior of an entity within a protocol. What is meaningful depends on the specific protocol (application), and on the entity that evaluates the trust relation. The application determines the exact semantics of trust, and the entity determines how the trust relation will be used in the ensuing steps of the protocol.

For example, suppose that entity A wants to determine the public key that entity B controls. A and B have had no previous interactions, hence no trust relation, so A has to contact entities that have some evidence about B's key. Relevant pieces of evidence in this case are certificates binding B's key to B's identity. Also, the trustworthiness of the entities that issued these certificates should be taken into account. If A has had previous interactions with these issuing entities then their public keys as well as their trustworthiness will be known to A. Otherwise, the same steps will have to be repeated to establish a trust relation with the issuing entities, recursively. Finally, A will evaluate the whole body of evidence and establish a trust relation with B. In this case, the trust relation will be : "A does (or does not) believe that B's key is K_B".

The specification of admissible types of evidence, the generation, distribution, discovery and evaluation of trust evidence are collectively called Trust Establishment. In this work, we are focusing on the evaluation process of trust evidence in Ad-Hoc Networks, i.e. we are focusing on the trust metric itself. In particular, we are not dealing with the collecting of evidence from the network, and the accompanying communication and signaling overhead. This issue is important, and obviously needs to be addressed in a complete system.

We will be using the terms "trust evaluation", "trust computation", and "trust inference" interchangeably. The evaluation process is formulated as a path problem on a weighted, directed graph. In this graph, nodes represent users, and edges represent direct trust relations, weighted by the amount of trust that the first user places on the second. Each user has direct relations only towards the users he has interacted with, so all interactions are local (in the trust graph). The aim is to establish an indirect relation between two users that have not previously interacted; this is achieved by using the direct trust relations that intermediate nodes have with each other. Hence, we assume that trust is transitive, but in a way that takes into account edge weights, too.

Ad Hoc networks are envisioned to have dynamic, sometimes rapidly-changing, random, multihop topologies which are likely composed of relatively bandwidth-constrained wireless links. The nodes themselves form the network routing infrastructure in an ad hoc fashion [6]. Based on these characteristics, we are imposing the following two main constraints on our scheme:

First, there is no preestablished infrastructure. The computation process cannot rely on, e.g., a Trusted Third Party. There is no Public Key Infrastructure, Certification Authorities, or Registration Authorities with elevated privileges.

Second, evidence is uncertain and incomplete. Evidence is generated by the users on the fly, without lengthy processes. So, it is uncertain. Furthermore, in the presence of adversaries, we cannot assume that all friendly nodes will be reachable: the malicious users may have rendered a small or big part of the network unreachable.

We require that the results are as accurate as possible, yet robust in the presence of attackers. It is desirable to, for instance, identify all allied nodes, but it is even more desirable that no adversary is misidentified as good. We use a general framework for path problems on graphs as a mathematical basis for our proposed scheme, and also give intuitive requirements that any trust evaluation algorithm should have under that framework. We evaluate the performance of the scheme with simulations on various trust topologies.

This work is organized in five sections. After this Introduction, the second section describes and comments on related work that has been done in the field of trust computation. The third section explains our approach, proposes a mathematical framework for trust computation, and describes intuitive properties that any scheme under this framework should have. In the fourth section, our proposed scheme is used for actual computation scenarios, and the results are discussed. The fifth section concludes the paper and suggests future directions for improvement.

2. RELATED WORK

Some of the following examples have been cast in the public key certification framework, whereas others are more general. However, they can all be viewed as trust evaluation metrics, insofar as they can compute an expected validity level for a statement like "Is this public key certificate valid?". In all work described below, graphs are used to model the situation at hand. The nodes correspond to entities, and an edge (i, j) is labeled according to the parameters of the certificate (credentials) issued by entity i for entity j.

Blaze, Feigenbaum, and Lacy [2] seem to have been the first to introduce the term "Trust Management", and identify it as a separate component of security services in networks. They designed and implemented the PolicyMaker trust management system, which provides a unified framework for describing policies (rules), credentials (trust evidence), and trust relationships. Also, this system is locally controlled, since it does not rely on a centralized authority to evaluate the credentials: Each user has the freedom and the responsibility to reach his own decisions.

The main issues in [2] and related work (KeyNote [1], SPKI/SDSI [5], Delegation Logic [18], Trust Policy Language [12], also [27]) are: the language in which the credentials and the policies will be described; the compliance-checking algorithm that checks if the credentials satisfy the policy rules; and the algorithm for the discovery of the credentials in the first place.

In PGP [28], a distinction is made between the *validity* of a public key and its *trust level*. Bob's key is valid for Alice, if Alice believes that it really belongs to Bob. The trust level of Bob's key corresponds to how carefully Bob authenticates keys before issuing certificates for them. The trust levels of the keys known to Alice are assigned in any way Alice wants. PGP only determines the validity of a key according to how many keys have signed it, and what the trust levels of the signing keys are. For instance, a key is valid if at least two marginally trusted keys have signed it.

Maurer's metric [21],[16] assigns weights $w_{ij} \in [0, 1]$ to edges. These weights correspond to i's opinion about the trustworthiness of the certificate issued for j's public key, i.e. to what degree i believes that the {public key – owner ID} binding implied by the edge $i \to j$ has been properly certified. The weights are then treated as link survival probabilities. The metric calculates the probability that at least one path survives that leads from the entity evaluating the metric to the entity involved in the certificate in question.

Reiter and Stubblebine [23] introduced the concept of path independence for entity authentication. They argued that multiple independent paths are a safer way to authenticate Bob than either the reachability or the bounded reachability metric. Their proposal was a Bounded Length Independent Paths metric which returns a set of node-disjoint paths from Alice (source) to Bob (destination) which are shorter than K hops. Since the computation of this metric is an NP-complete problem for $K \geq 5$ they gave approximation algorithms.

In a subsequent paper [24], the same people suggested a different metric, based on network flow techniques. The model being the same, weights were added on the edges indicating the amount of money that the issuer will pay to anyone who is misled because of the corresponding certificate is false. Treating the edge weights as capacities, the metric calculates the maximum flow from Alice to Bob. This is the minimum amount of money that Alice can expect to receive if it turns out that she has been using the wrong key for Bob.

Levien's metric [17] is also network flow based. After assigning edge capacities the metric treats trust as a commodity that flows from Alice to Bob. Alice has unit quantity of trust and tries to send it to Bob. The metric calculates how much of this unit quantity reaches Bob. By suitably assigning capacities, the metric is made more resistant to attacks. However, some assumptions in this work are not realistic, e.g. that all nodes have the same indegree d.

Jøsang [14] has developed an algebra for assessing trust relations, and he has applied it to certification chains. To a statement like "The key is authentic" he is assigning a triplet (called *opinion*) $(b, d, u) \in [0, 1]^3 : b + d + u = 1$, where b, d, and u designate belief, disbelief, and uncertainty respectively. Belief (disbelief) in a statement increases when supporting (contradicting) evidence appears. Uncertainty is caused by the lack of evidence to support either belief or disbelief. Many operators are given for the manipulation of these opinions.

In [3], first-hand observations are locally exchanged between neighboring nodes. Assume i receives from j evidence about k. First of all, i adjusts his opinion for j, based on how close j's evidence is to i's previous opinion about k. If it is not closer than some threshold, the new evidence is discarded, and i lowers his opinion about j. Otherwise, i increases his trust for j and the new evidence is merged with i's existing opinion for k.

In [19], a group Q of users is selected, and they are asked to give their opinion about a certain target node. The end result is a weighted average of their opinions and any preexisting opinion that the initiator node may have. One possible selection for the group Q is the one-hop neighbors of the initiator.

In the EigenTrust algorithm [15], nodes exchange vectors of personal observations (called *local trust values*) with their one-hop neighbors. Node i's local trust value for node j is denoted by c_{ij}. These trust values are normalized ($\forall i : \sum_j c_{ij} = 1$). Each node i calculates global trust values t_{ij} for all other nodes j by the following iterative computation: $t_{ij}^{(n+1)} = \sum_k c_{ik} t_{kj}^{(n)}$, where $t_{kj}^{(0)} = c_{kj}$.

3. OUR APPROACH

3.1 System Model

We view the trust inference problem as a generalized shortest path problem on a weighted directed graph $G(V, E)$ (*trust graph*). The vertices of the graph are the users/entities in the network. A weighted edge from vertex i to vertex j corresponds to the *opinion* that entity i, also referred to as the *issuer*, has about entity j, also referred to as the *target*. The weight function is $w(i, j) : V \times V \longrightarrow S$, where S is the opinion space. The aim is to aggregate, in a meaningful way, the weights along all paths from the source to the destination.

Each opinion consists of two numbers: the *trust* value, and the *confidence* value. The former corresponds to the issuer's estimate of the target's trustworthiness. For example, a high trust value may mean that the target is one of the good guys, or that the target is able to give high quality location information, or that a digital certificate issued for the target's public key is believed to be correct. The confidence value corresponds to the accuracy of the trust value assignment. A high confidence value means that the target has passed a large number of tests that the issuer has set, or that the issuer has interacted with the target for a long time. Since opinions with a high confidence value are more useful in making trust decisions, the confidence value is also referred to as the *quality* of the opinion. The space of opinions can be visualized as a rectangle (MIN_T, MAX_T)×(MIN_C, MAX_C) in the Cartesian plane (Figure 1, for $S = [0, 1] \times [0, 1]$).

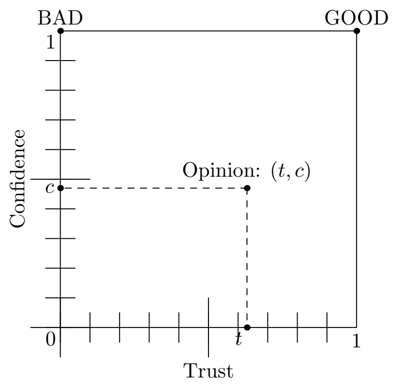

Figure 1: Opinion space

Both the trust and the confidence values are assigned by the issuer, in accordance with his own criteria. This means that a node that tends to sign public key certificates without too much consideration will often give high trust and high confidence values. The opposite holds true for a strict entity. When two such entities interact, it is important for the stricter entity to assign a low enough trust value to the less strict one, indicating the perceived "sloppiness". Otherwise, the less strict entity may lead the stricter one to undesirable trust decisions. This situation is easier to picture in the context of Certification Authorities and public key certification. There, a CA A will only give a high trust value to B, if B's policy for issuing certificates is at least as strict as A's and has the same durability characteristics [11].

Also, it is assumed that nodes assign their opinions based on local observations. For example, each node may be equipped with a mechanism that monitors neighbors for evidence of malicious behavior, as in [20]. Alternatively, two users may come in close contact and visually identify each other, or exchange public keys, as suggested in [4]. In any case, the input to the system is local: however, extant pieces of evidence based on, e.g., previous interactions with no longer neighboring nodes can also be taken into account for the final decision. This would come into play when two nodes that have met in the past need now to make a trust decision for each other. Of course, the confidence value for such evidence would diminish over time. One consequence of the locality of evidence gathering is that the trust graph initially overlaps with the physical topology graph: The nodes are obviously the same, and the edges are also the same if the trust weights are not taken into account. As nodes move, opinions for old neighbors are preserved, so the trust graph will have more edges than the topology graph. However, as time goes by, these old opinions fade away, and so do the corresponding edges.

In the framework described, two versions of the trust inference problem can be formalized. The first is finding the trust-confidence value that a source node A should assign to a destination node B, based on the intermediate nodes' trust-confidence values. Viewed as a generalized shortest path problem, it amounts to finding the generalized distance between nodes A and B. The second version is finding the most trusted path between nodes A and B. That is, find a sequence of nodes $\langle v_0 = A, v_1, \ldots, v_k = B\rangle : (v_i, v_{i+1}) \in E, 0 \le i \le k-1$ that has the highest aggregate trust value among all trust paths starting at A and ending at B. A high level view of the system is shown in Figure 2.

Both problems are important: finding a target's trust value is needed before deciding whether to grant him access to one's files, or whether to disclose sensitive information, or what kind of orders he is allowed to give (in a military scenario, for instance). In this case, we will usually utilize multiple trust paths to find the trust distance from the source to the destination, since that will increase the evidence on which the source bases its final estimate. Consequently, there may not be a single "best" path chosen to reach this estimate.The second problem is more relevant when it comes to actually communicating with a target node. The target node being trustworthy is one thing, but finding a trusted path of nodes is needed, so that traffic can be routed through them. In this case, we are interested in finding only one path. This will, in a sense, be the "best" path available.

The core of our approach is the two operators that are used to combine opinions: One operator (denoted $\otimes$) com-

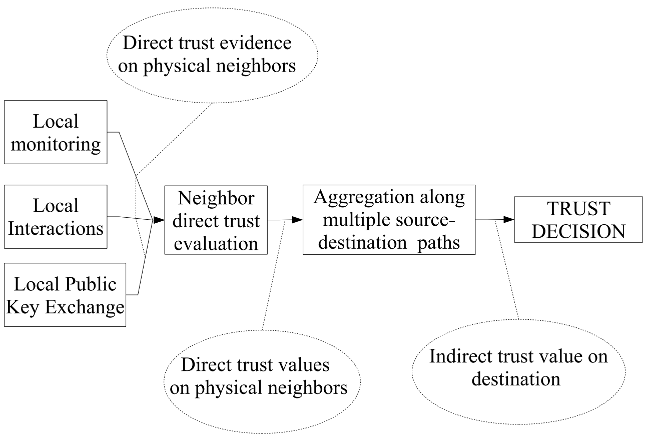
Direct trust evidence on physical neighbors
Local monitoring
Local Interactions
Local Public Key Exchange
Neighbor direct trust evaluation
Aggregation along multiple source-destination paths
TRUST DECISION
Direct trust values on physical neighbors
Indirect trust value on destination

Figure 2: System operation

bines opinions along a path, i.e. A's opinion for B is combined with B's opinion for C into one indirect opinion that A should have for C, based on B's recommendation. The other operator (denoted $\oplus$) combines opinions across paths, i.e. A's indirect opinion for X through path p_1 is combined with A's indirect opinion for X through path p_2 into one aggregate opinion that reconciles both. Then, these operators can be used in a general framework for solving path problems in graphs, provided they satisfy certain mathematical properties, i.e. form an algebraic structure called a semiring. More background on this general framework is in section 3.2. The operators are discussed in greater depth in section 3.3.

3.2 Semirings

For a more complete survey of the issues briefly exposed here, see [25].

3.2.1 Definitions

A *semiring* is a triplet $(S, \oplus, \otimes)$, where S is a set, and $\oplus, \otimes$ are binary operators with the following properties ($a, b, c \in S$):

- $\oplus$ is commutative, associative, with a neutral element ⓪ $\in S$
- $\otimes$ is associative, with a neutral element ① $\in S$, and ⓪ as an absorbing element
- $\otimes$ distributes over $\oplus$

For example, the set of real numbers, along with the usual addition and multiplication, forms a semiring. The neutral elements obviously are ⓪ $\equiv 0$, and ① $\equiv 1$.

A semiring $(S, \oplus, \otimes)$ with a partial order relation $\preceq$ that is monotone with respect to both operators is called an *ordered semiring* $(S, \oplus, \otimes, \preceq)$:

$$a \preceq b \text{ and } a' \preceq b' \Longrightarrow a \oplus a' \preceq b \oplus b' \text{ and } a \otimes a' \preceq b \otimes b'$$

A semiring is called idempotent when the following holds:

$$\forall a \in S : a \oplus a = a$$

3.2.2 Semirings for path problems

In the context of the generalized shortest path problem in a weighted graph, $\otimes$ is the operator used to calculate the weight of a path based on the weights of the path's edges:

$$p = (v_0, v_1, \ldots, v_k),$$
$$w(p) = w(v_0, v_1) \otimes w(v_1, v_2) \otimes \cdots \otimes w(v_{k-1}, v_k)$$

The $\oplus$ operator is used to compute the shortest path weight d_{ij} as a function of all paths from the source i to the destination j:

$$d_{ij} = \bigoplus_{\substack{p \text{ is a path} \\ \text{from } i \text{ to } j}} w(p)$$

In the familiar context of edge weights being transmission delays, the semiring used is $(\Re_+ \cup \{\infty\}, \min, +)$, i.e. $\oplus$ is min, and $\otimes$ is $+$: The total delay of a path is equal to the sum of all constituent edge delays, whereas the shortest path is the one with minimum delay among all paths. Also, ⓪ is ∞, and ① is 0. On the other hand, if edge weights are link capacities, then the maximum bottleneck capacity path is found by the semiring $(\Re_+ \cup \{\infty\}, \max, \min)$, with ⓪ $\equiv 0$, ① $\equiv \infty$. The transitive closure of a graph uses the Boolean semiring: $(\{0, 1\}, \vee, \wedge)$, where all edge weights are equal to 1. This answers the problem of path existence.

Note that the $\oplus$ operator may pick a single path weight (as is the case with max and min) or it may explicitly combine information from all paths (e.g., addition, averaging).

3.3 Trust Semiring

3.3.1 Intuitive Requirements

Based on intuitive concepts about trust establishment, we can expect the binary operators to have certain properties in addition to those required by the semiring structure.

Since an opinion should deteriorate along a path, we require the following for the $\otimes$ operator ($a, b \in S$):

$$a \otimes b \preceq a, b$$

where $\preceq$ is the difference relation defined in Section 3.2. Note that the total opinion along a path is "limited" by the source's opinion for the first node in the path.

The element ⓪ (neutral element for $\oplus$, absorbing for $\otimes$) is the set of opinions (t, MIN_C), for any $t \in [0, 1]$, which, in essence, corresponds to non-existent trust relations between nodes. The motivation is that if a ⓪ is encountered along a path, then the whole path "through" this opinion should have zero confidence. Also, such opinions should be ignored in $\oplus$-sums.

The element ① (neutral element for $\otimes$) is the "best" opinion that can be assigned to a node: (MAX_T, MAX_C). This can be seen as the opinion of a node about itself. Also, it is the desirable point of convergence of the opinions of all good nodes about all other good nodes in the classification example. If encountered along a path, ① effectively contracts the corresponding edge and identifies the nodes at its endpoints for the purposes of the aggregation.

Regarding aggregation across paths with the $\oplus$ operator, we generally expect that opinion quality will improve, since we have multiple opinions. If the opinions disagree, the more confident one will weigh heavier. If we insist in picking exactly one opinion, we come up against a problem in the, admittedly rare, case when the opinions have equal confidence values, but different trust values. In that case we have to make a conscious design decision to choose the higher trust value (optimistic) or the lower trust value (pessimistic). In a fashion similar to the $\otimes$ operator, we require that the $\oplus$ operator satisfies $(a, b \in S)$:

$$a \oplus b \succeq a, b$$

3.3.2 Path semiring

In this semiring, the opinion space is $S = [0,1] \times [0,1]$ Our choice for the $\otimes$ and $\oplus$ operators is as follows (Figure 3):

$$(t_{ik}, c_{ik}) \otimes (t_{kj}, c_{kj}) = (t_{ik}t_{kj}, c_{ik}c_{kj}) \quad (1)$$

$$(t_{ij}^{p_1}, c_{ij}^{p_1}) \oplus (t_{ij}^{p_2}, c_{ij}^{p_2}) = \begin{cases} (t_{ij}^{p_1}, c_{ij}^{p_1}) & \text{if } c_{ij}^{p_1} > c_{ij}^{p_2} \\ (t_{ij}^{p_2}, c_{ij}^{p_2}) & \text{if } c_{ij}^{p_1} < c_{ij}^{p_2} \\ (\max(t_{ij}^{p_1}, t_{ij}^{p_2}), c_{ij}^{p_1}) & \text{if } c_{ij}^{p_1} = c_{ij}^{p_2} \end{cases} \quad (2)$$

where $(t_{ij}^{p_1}, c_{ij}^{p_1})$ is the opinion that i has formed about j along the path p_1.

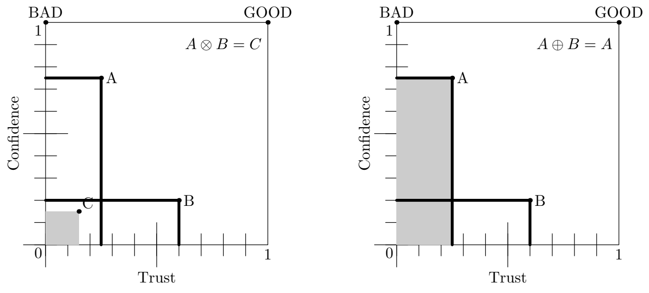

Figure 3: $\otimes$ and $\oplus$ operators for the Path semiring

Since both the trust and the confidence values are in the $[0,1]$ interval, they both decrease when aggregated along a path. When opinions are aggregated across paths, the one with the highest confidence prevails. If the two opinions have equal confidences but different trust values, we pick the one with the highest trust value. We could have also picked the lowest trust value; the choice depends on the desired semantics of the application.

This semiring essentially computes the trust distance along the most confident trust path to the destination. An important feature is that this distance is computed along a single path, since the $\oplus$ operator picks exactly one path. Other paths are ignored, so not all available information is being taken into account. One of the advantages is that if the trust value turns out to be high, then a trusted path to the destination has also been discovered. Also, fewer messages are exchanged for information gathering.

3.3.3 Distance semiring

Our second choice is a semiring based on the *Expectation semiring* which was defined by Eisner in [9], and was applied in the field of language and speech processing. The opinion space is $S = [0, \infty] \times [0, 1]$. Before using this semiring, the pair (trust, confidence)=(t, c) is mapped to the weight $(c/t, c)$. The motivation for this mapping becomes clear when we describe its effect on the results of the operators. The binary operators are then applied to this weight, and the result is mapped back to a (trust, confidence) pair. The whole process is displayed in Equations 3 and 4, where arrows denote mappings, and "equals" signs denote actual calculations based on the semiring operators.

$$\begin{aligned} (t_{ik}, c_{ik}) \otimes (t_{kj}, c_{kj}) &\rightarrow \left(\frac{c_{ik}}{t_{ik}}, c_{ik}\right) \otimes \left(\frac{c_{kj}}{t_{kj}}, c_{kj}\right) \\ &= \left(\frac{c_{ik}c_{kj}}{t_{ik}} + \frac{c_{ik}c_{kj}}{t_{kj}}, c_{ik}c_{kj}\right) \\ &\rightarrow \left(\frac{1}{\frac{1}{t_{ik}} + \frac{1}{t_{kj}}}, c_{ik}c_{kj}\right) \end{aligned} \quad (3)$$

$$\begin{aligned} (t_{ij}^{p_1}, c_{ij}^{p_1}) \oplus (t_{ij}^{p_2}, c_{ij}^{p_2}) &\rightarrow \left(\frac{c_{ij}^{p_1}}{t_{ij}^{p_1}}, c_{ij}^{p_1}\right) \oplus \left(\frac{c_{ij}^{p_2}}{t_{ij}^{p_2}}, c_{ij}^{p_2}\right) \\ &= \left(\frac{c_{ij}^{p_1}}{t_{ij}^{p_1}} + \frac{c_{ij}^{p_2}}{t_{ij}^{p_2}}, c_{ij}^{p_1} + c_{ij}^{p_2}\right) \\ &\rightarrow \left(\frac{c_{ij}^{p_1} + c_{ij}^{p_2}}{\frac{c_{ij}^{p_1}}{t_{ij}^{p_1}} + \frac{c_{ij}^{p_2}}{t_{ij}^{p_2}}}, c_{ij}^{p_1} + c_{ij}^{p_2}\right) \end{aligned} \quad (4)$$

So (Figure 4), when aggregating along a path, both the trust and the confidence decrease. The component trust values are combined like parallel resistors. We can see here the effect of the mapping: Two resistors in parallel offer lower resistance than either of them in isolation. Also, a zero trust value in either opinion will result in a zero trust value in the resulting opinion (absorbing element), while a trust value equal to infinity will cause the corresponding opinion to disappear from the result (neutral element). On the other hand, the component confidence values are between 0 and 1, and they are multiplied, so the resulting confidence value is smaller than both.

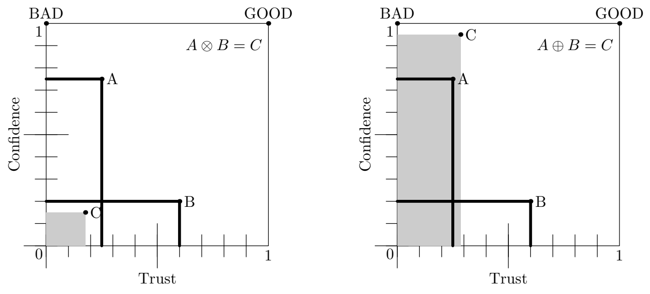

Figure 4: $\otimes$ and $\oplus$ operators for the Distance semiring

When aggregating across paths, the total trust value is the weighted harmonic average of the components, with weights proportional to their confidence values. So, the result is something between the two component values, but closer to the more confident one. Again we can see the effect of the mapping: The weighted harmonic average outcome is a direct result of the inverse mapping. Note, also, the behavior caused by extreme (zero or infinity) trust values: A zero trust value dominates the result (unless its corresponding confidence is zero); a trust value equal to infinity results in an increase in the trust value given by the other opinion. In order for the resulting trust value to be the maximum possible, both opinions have to assign the maximum. So, in general, we can say that this operator is conservative. A zero confidence value (neutral element) causes the corresponding opinion to have no effect on the result.

3.3.4 Computation algorithm

The following algorithm, due to Mohri [22], computes the $\oplus$-sum of all path weights from a designated node s to all other nodes in the trust graph $G = (V, E)$.

```
GENERIC-SINGLE-SOURCE-SHORTEST-DISTANCE($G, s$)
1   for $i \leftarrow 1$ to $|V|$
2       do $d[i] \leftarrow r[i] \leftarrow \textcircled{0}$
3   $d[s] \leftarrow r[s] \leftarrow \textcircled{1}$
4   $S \leftarrow \{s\}$
5   while $S \neq \emptyset$
6       do $q \leftarrow head(S)$
7          DEQUEUE($S$)
8          $r' \leftarrow r[q]$
9          $r[q] \leftarrow \textcircled{0}$
10         for each $v \in Neighbors[q]$
11             do if $d[v] \neq d[v] \oplus (r' \otimes w[(q, v)])$
12                   then $d[v] \leftarrow d[v] \oplus (r' \otimes w[(q, v)])$
13                        $r[v] \leftarrow r[v] \oplus (r' \otimes w[(q, v)])$
14                        if $v \notin S$
15                           then ENQUEUE($S, v$)
16  $d[s] \leftarrow \textcircled{1}$
```

This is an extension to Dijkstra's algorithm [7]. S is a queue that contains the vertices to be examined next for their contribution to the shortest path weights. The vector $d[i], i \in V$ holds the current estimate of the shortest distance from s to i. The vector $r[i], i \in V$ holds the total weight added to $d[i]$ since the last time i was extracted from S. This is needed for non-idempotent semirings, such as the second one proposed.

Our computation algorithm is based on Mohri's, but with three adjustments which are needed when considering the problem from the perspective of trust. Lines 11-13 of the algorithm will be referred to as "node q votes for node v".

First of all, some nodes may be prevented from voting. Only if a node's trust value exceeds a predefined trust threshold, is the node allowed to vote. This is motivated from the common sense observation that only good nodes should participate in the computation, and bad nodes should be barred. Note that there is no restriction on the corresponding confidence. Therefore, bad nodes will initially be allowed to vote, but after some point they will be excluded since good nodes will acquire evidence for their maliciousness.

Second, no node is allowed to vote for the source (s). Since it is s that initiates the computation, it does not make sense to compute s's opinion for itself.

Third, no cyclic paths are taken into account. If that were the case, we would be allowing a node to influence the opinion about itself, which is undesirable. Unfortunately, there is no clear way to discard any single edge-opinion of the cycle. So, the approach taken is to discard any edges that would form a cycle if accepted. As a result, the order in which the voters are chosen in line 6 is important. We choose the node for which the confidence is highest.

Note that these adjustments introduce characteristics from the Path semiring into the Distance semiring. For example, the node with the maximum confidence gets to vote first. Moreover, some paths are pruned which means that fewer messages are exchanged, thus saving bandwidth, but also some of the existing information is not taken into account. In general, this combination of the two semirings seems to be a good tradeoff between the two.

3.4 Comparison to related work

Quantitative comparison to similar research is hard, since there is no common framework for objective measurements. For example, the notion of confidence is only in two of the references (discussed below), but it occupies a central position in our work, and we indeed consider it an advantage of our work since it makes for a more expressive model. There is no obvious way to transform weights across schemes. We used these schemes to derive our intuitive trust requirements, and avoid pitfalls like allowing a node to single-handedly increase other nodes' opinion about himself.

In [3] opinions are modeled as probability distribution functions (pdfs), so the uncertainty (inverse of confidence) in an opinion corresponds to the variation of the pdf. However, there are many other parameters (behavior model of Bad and Good nodes, node mobility, local vs global opinion availability) that obstruct a meaningful comparison without oversimplifications.

In [14] uncertainty is also modeled explicitly, and in fact this work is the closest to ours since it is also based on an algebraic framework for evaluating trust. Unfortunately, their operators do not satisfy the distributivity property, the end result of which is that some trust graphs cannot be analysed, or can be analysed in more than one way with different results. Additionally, the lack of distributivity means that the existing theory of semirings and the associated results can no longer be used. Their supporting argument (that only independent opinions should be aggregated, see [14] for more details) does not sound 100% intuitive. This should rather be considered an open issue, and subject for further investigation. Another benefit of distributivity is that it allows in-network aggregation of opinions, which is a way of saving bandwidth. It also allows incremental incorporation of new paths, which is helpful in a distributed, asynchronous environment. All criticism aside, this work was a big initial inspiration for the discovery of our trust semiring framework.

It transpires that a common ground for performing meaningful comparisons is direly needed. Hopefully, our semiring formalism (or something that subsumes it) will turn out to be expressive enough to serve that purpose.

4. EVALUATION AND EXPERIMENTAL RESULTS

In this section, we are describing the scenarios that were examined in the simulations. The results obtained are discussed, and explained in terms of the parameters and properties of the algorithms.

4.1 Simulation setup

We assume that some nodes are Good, and some are Bad. Good nodes adjust their direct opinions (opinions for their neighbors) according to some predefined rules (explained later in this section). Bad nodes, however, always have the best opinion (MAX_T,MAX_C) for their neighboring Bad nodes, and the worst opinion (MIN_T,MAX_C) for their neighboring Good nodes. This is a rather simplistic strategy, and we plan to make it more elaborate in future work.

We expect that the opinions of a Good node for all other nodes would evolve as in Figure 5. That is, all Good and all Bad nodes will be identified as Good and Bad, respectively. When the network is "born", the nodes are partitioned into

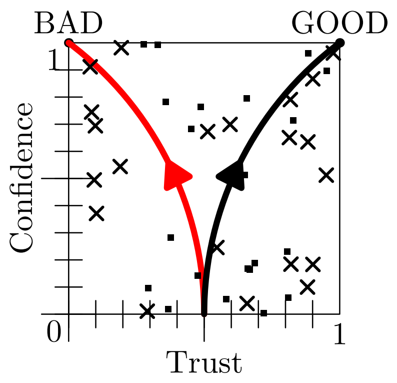

Figure 5: Opinion convergence. Opinions for good and bad nodes are denoted with a cross and a square, respectively.

Good and Bad. We pick a Good node, which will be computing indirect opinions to all other nodes. Initial direct opinions are all set to values randomly distributed around $(0.5, 0.1)$, i.e. medium trust and low confidence. The trust threshold is empirically set to 0.3: Only nodes for which the trust value is higher than this threshold are allowed to vote. Time is discrete and is measured in rounds. At each round, two things happen. First, the direct opinions of each node for his neighbors approach the correct opinion, which is (MIN_T,MAX_C) for the bad neighbors, and (MAX_T,MAX_C) for the good neighbors. Second, the designated good node calculates his indirect opinions for all other nodes. These indirect opinions are the experimental results shown in Section 4.2. Also, the confidence for some indirect opinions may be too low (within $\epsilon = 0.01$ of MIN_T $= 0$), so these nodes are not assigned any opinion. Note that plenty of real time may pass between two successive rounds. This real time amounts to communication delay due to opinion exchanging between the nodes. We are abstracting away this delay, as well as all the specifics of the exchanging algorithm, and we are leaving it for future work.

The most important evaluation metric is whether the nodes are correctly classified as good and bad. We want the opinions for all bad nodes to be close to (MIN_T,MAX_C) and the opinions for all good nodes close to (MAX_T,MAX_C). Moreover, we want this to happen as soon as possible, i.e. before all direct opinions converge to the correct ones, since the users in the real network may be forced to make an early trust decision. Furthermore, a failsafe is desirable: If trust evidence is insufficient, we prefer not to make any decision about a node, rather than make a wrong one.

Of course, we have to evaluate the robustness of each of the above mentioned metrics as the proportion of bad nodes increases. We also measure the effect of different trust topologies. Namely, three topologies are selected: *Grid*, *Random*, and *Small World*. The *Grid* and *Random* topologies can be seen as two extremes of a spectrum. On the one hand, the Grid is completely symmetric and deterministic: We are using a 10x10 square for 100 nodes. Each node, except the perimeter nodes, has exactly 8 neighbors. On the other hand, the Random topology was constructed so that the average degree is again 8, but this symmetry is completely probabilistic. Each edge has the same probability of existing, according to the Erdös-Rényi model [10]. The Small World topology [26] is between these two extremes, in the sense that there are a few nodes that have a high degree, and all the rest have much fewer neighbors. In this case, too, the average degree is 8. The Small World topology for trust has also been used in [13].

4.2 Results

In this section we present some of the results obtained through simulations. We chose to present few actual output data, and more detailed conclusions. For each of the three topologies (Grid, Random, Small World), the percentage of bad nodes is increased from 10% to 50% to 90%. The example figure below (Figure 6) shows the opinions of the source node (s) for every other node after the computations of rounds 30 and 70 in the Random topology for a 50% percentage of Bad nodes. The nodes originally designated as Good are pictured as crosses, whereas the Bad ones as squares. The aim is, first and foremost, for the two groups of shapes to be clearly separated. Also, the Good nodes should be as close as possible to the upper right corner (GOOD corner, corresponding to the (MAX_T,MAX_C) opinion), and the Bad nodes to the upper left corner (BAD corner, (MIN_T,MAX_C) opinion).

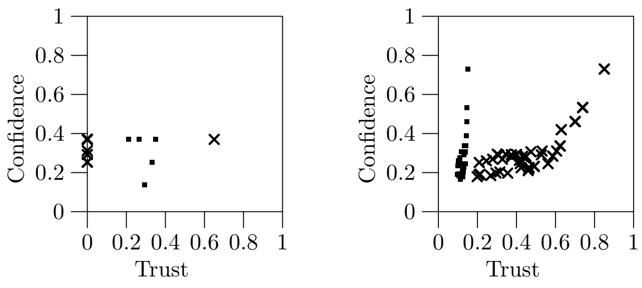

Figure 6: Rounds 30 and 70, Good nodes are crosses, Bad nodes are squares

4.3 Discussion

We were able to observe some general trends in the results obtained. First of all, in the early rounds Good and Bad nodes are intermixed: there is no clear separating line. Even more, Bad nodes seem to be given better opinions than Good nodes, which is clearly undesirable. The explanation for this is based on two aspects of the scheme; namely, the trust threshold and the Bad nodes' way of assigning direct opinions. Initially, Bad nodes are allowed to vote, since the trust threshold (0.3) is lower than the initial default trust value (0.5), i.e. they have not been "discovered" yet. So, their (MIN_T,MAX_C)=(0,1) opinions for Good nodes are taken into account and the result is that Good nodes appear to be bad. Also, Bad nodes give (MAX_T,MAX_C)=(1,1) opinions to each other, hence reinforcing each other.

The situation in later rounds improves. The Good nodes move towards the upper right corner, the Bad ones towards the upper left. There is also a clear separating line between the two groups of nodes. For an actual implementation a practical guideline could be derived from the above observation, i.e. to be especially careful when making important trust decisions in early rounds. The trust computation may be based on too little raw evidence (direct opinions) to be relied upon. In all cases, however, the Good and Bad nodes are separated eventually (in the last rounds). This serves as a sanity check for the algorithm.

As the percentage of Bad nodes increases, we can see that the separation is still successful sooner or later, but the main observation is that the number of classified nodes is decreasing, especially for the Grid topology. Classified nodes are those for which the evidence was sufficient, i.e. the confidence of the source's opinion for them was more than $\epsilon = 0.01$. The following graphs (Figure 7) show the number of nodes classified in each topology, for different percentages of Bad nodes, after every round of computation. The general effect of Bad nodes on the number of classified nodes is that, after they are discovered, they block the trust paths they are on since they are not allowed to vote. So, nodes that are further away from the source than these Bad nodes can be reached by fewer paths. They may even be completely isolated. In any case, the confidence in the source's opinion for them is decreased, so some of them cannot be classified.

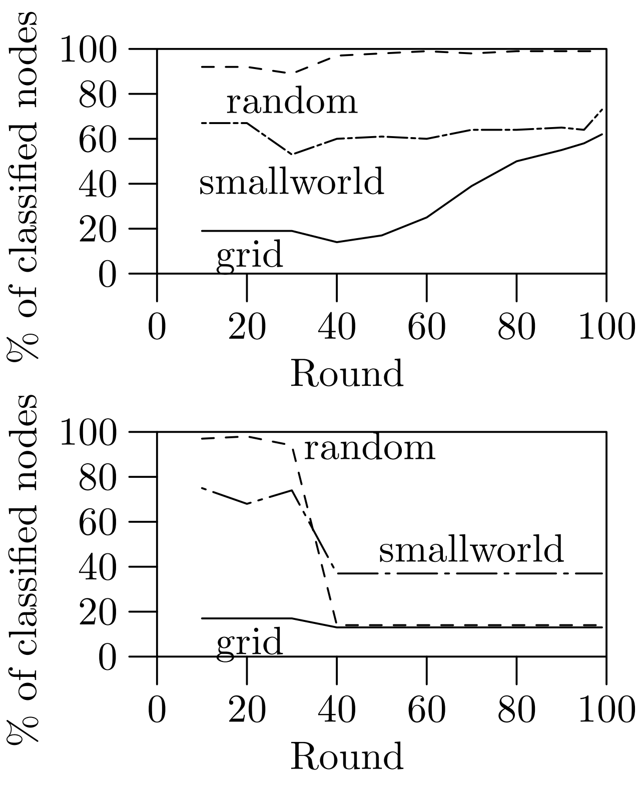

Figure 7: Node classification, 50% 90% bad nodes

The Random topology performs best, because it is less affected by Bad nodes. This topology has a relatively short average path length between the source (s) and all other nodes, so confidence values for opinions are not too low. At the same time, it does not rely on information provided by any single node or small set of nodes. The links are random, so every node is reached through different paths.

The average path length is the main defect of the Grid topology, since for certain nodes it is large. If this is coupled with Bad nodes blocking some of the paths, the confidence values for nodes that are away from the source is dropping considerably. The more bad nodes, the more pronounced this effect is. So, the Grid topology performs worst of all.

As far as the Small World topology is concerned, the path length is short, since there are some highly connected nodes. So, it performs better than the Grid topology. However, it is exactly these highly connected nodes that degrade the performance of the computation when they are Bad. The reason is, again, that they block many paths and affect opinions for most nodes. If the majority of these highly connected nodes are Bad, few trust paths will be able to be established.

The 90% Bad node case is interesting to examine specifically. First, there is a sudden drop in the number of classified nodes between rounds 30 and 40. This is so, because at this point the opinions for Bad nodes acquire trust values that are lower than the trust threshold, so they become ineligible to vote. This could suggest the Sybil attack [8] as an attempt to compromise the system. By introducing an arbitrary number of fake Bad nodes, one could arbitrarily increase their percentage. This seems to allow a Bad node to bring about the 90%(+) case at will, and therefore only a very small number of nodes will be classified. However, things are not so simple: The invented Bad nodes do not block any existing "good" trust paths. They may create additional "bad" paths, but these will contribute to the final result neither positively nor negatively: they will be ignored. What really matters is the number of "good" trust paths, and nothing else. In our previous discussion, we have been increasing the percentage of Bad nodes *keeping the total number of nodes constant*. So, we have been reducing the number of Good nodes, and, as a result, the number of "good" paths.

Second, and more intriguing, is that the Random topology becomes equivalent to the Grid topology, and the Small World topology performs better than both. The explanation is that almost all nodes are Bad, so only nodes one or two hops away from the source can be classified. This is true for all topologies. But the Grid nodes have exactly 8 neighbors, and all Random nodes have approximately 8 neighbors, too. So, the number of classified nodes turns out to be around 20. On the other hand, in the Small World topology the source node is one of the highly connected nodes (19 neighbors, when the average degree is 8). So, all of the 19 neighbors, and some of the nodes that are two hops away are classified for a total of about 40 nodes. A practical guideline for the Small World topology would then be that highly connected nodes should be protected, better prepared to withstand attacks, or, in general, less vulnerable.

5. CONCLUSION AND FUTURE WORK

We have presented a scheme for evaluating trust evidence in Ad-Hoc networks. Our scheme is entirely based on information originating at the users of the network. No centralized infrastructure is required, although the presence of one can certainly be utilized. Also, users need not have personal, direct experience with every other user in the network in order to compute an opinion about them. They can base their opinion on second-hand evidence provided by intermediate nodes, thus benefitting from other nodes' experiences. Of course, we are taking into account the fact that second-hand (or third, or fourth...) evidence is not as valuable as direct experience. In this sense, our approach extends PGP, since PGP only uses directly assigned trust values.

At each round of computation, the source node computes opinions for all nodes. This means that information acquired at a single round can be stored and subsequently used for many trust decisions. If there is not enough evidence to determine an opinion, then no opinion is formed. So, when malicious nodes are present in the network they cannot fool the system into accepting a malicious node as benevolent. A failsafe state exists that ensures graceful degradation as the number of adversaries increases.

The trust topology also has significant influence on the performance of the algorithm. We have seen that if any node can be malicious with the same probability, the Random topology performs better. On the other hand, if the highly connected nodes of the Small World topology are Good, the algorithm fares better at the crucial cases of malicious node preponderance.

One of the main advantages of our work is the explicit modeling of the uncertainty in trust estimates. This allows greater flexibility in describing intuitive situations. For example, it is possible to distinguish between the following opinions: "I am 100% confident that X is 50% trustworthy" and "I am 50% confident that X is 50% trustworthy". This distinction would not have been possible, if we keep just one trust value and increase (or decrease) it according to our observations. Two good and two bad observations would be equivalent to two hundred good and two hundred bad ones.

The use of a formal framework (semirings) and the associated treatment of trust evaluation (generalized path problem) is important in its own right. It allows transfer of knowledge from well developed theoretical areas, provides better insight into the design considerations of new trust evaluation schemes, and potentially enables analytical proofs of hard results or bounds.

In future work, we plan to implement more elaborate models for the attackers' behavior, and for the measures taken against nodes that are being assigned low trust values (i.e., detected to be bad). So, the attackers will be facing a trade-off between the amount of damage they can inflict, and the possibility of being, for instance, isolated from the rest of the network. Suitable strategies will be developed for Good as well as Bad nodes. Furthermore, an algorithm for gathering the evidence at the requesting node will be designed, and its communication and signaling costs evaluated. It is also interesting to study the effects of including some trust infrastructure, e.g. nodes for which trust would be fixed.

6. ACKNOWLEDGMENTS

This material is based on work done through collaborative participation in the Collaborative Technology Alliance for Communications & Networks sponsored by the U.S. Army Research Laboratory under Cooperative Agreement DAAD19-01-2-0011. Also, it was supported by the U.S. Army Research Office under Award No. DAAD19-01-1-0494. We would also like to thank the anonymous reviewers for their helpful comments. The Sybil attack discussion in Section 4.3 was motivated by one of these comments, and parts of the intended future research were suggested by reviewers. Many clarifications were made.

7. REFERENCES

[1] M. Blaze, J. Feigenbaum, J. Ioannidis, and A. D. Keromytis. The KeyNote trust management system. RFC 2704, Sept. 1999.

[2] M. Blaze, J. Feigenbaum, and J. Lacy. Decentralized trust management. In *Proceedings of the 1996 IEEE Symposium on Security and Privacy*, pages 164–173, 1996.

[3] S. Buchegger and J. Y. Le Boudec. The effect of rumor spreading in reputation systems for mobile ad-hoc networks. In *Proceedings of WiOpt '03: Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, Sophia-Antipolis, France, March 2003.

[4] S. Čapkun, J.-P. Hubaux, and L. Buttyán. Mobility helps security in ad hoc networks. In *Proceedings of the ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHOC 2003)*, Annapolis, MD, June 2003.

[5] D. Clarke, J.-E. Elien, C. Ellison, M. Fredette, A. Morcos, and R. L. Rivest. Certificate chain discovery in SPKI/SDSI. *Journal of Computer Security*, 9(4):285–322, 2001.

[6] S. Corson and J. Macker. Mobile ad hoc networking (manet): Routing protocol performance issues and evaluation considerations. RFC 2501, IETF, January 1999.

[7] E. W. Dijkstra. A note on two problems in connection with graphs. *Numerische Mathematik*, 1:269–271, 1959.

[8] J. R. Douceur. The sybil attack. In *Proceedings of the IPTPS02 Workshop*, Cambridge, MA, March 2002.

[9] J. Eisner. Parameter estimation for probabilistic finite-state transducers. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, Philadelphia, July 2002.

[10] P. Erdös and A. Rényi. On random graphs. *Publicationes Mathematicae*, 6:290–297, 1959.

[11] L. Eschenauer, V. D. Gligor, and J. Baras. On trust establishment in mobile ad-hoc networks. In B. Christianson, B. Crispo, J. A. Malcolm, and M. Roe, editors, *10th International Security Protocols Workshop, Cambridge, UK, April 2002*, volume 2845 of *Lecture Notes in Computer Science*, pages 47–66. Springer-Verlag, 2004.

[12] A. Herzberg, Y. Mass, J. Michaeli, D. Naor, and Y. Ravid. Access control meets public key infrastructure, or: Assigning roles to strangers. In *Proceedings of the 2000 IEEE Symposium on Security and Privacy*, pages 2–14, Berkeley, CA, May 2000.

[13] J.-P. Hubaux, L. Buttyán, and S. Čapkun. The quest for security in mobile ad hoc networks. In *Proceedings of the ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHOC 2001)*, 2001.

[14] A. Jøsang. An algebra for assessing trust in certification chains. In *Proceedings of the Network and Distributed Systems Security (NDSS'99) Symposium*, 1999.

[15] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina. The EigenTrust algorithm for reputation management in p2p networks. In *WWW2003*, May 2003.

[16] R. Kohlas and U. Maurer. Confidence valuation in a public-key infrastructure based on uncertain evidence. In *Proceedings of Public Key Cryptography 2000*, volume 1751 of *Lecture Notes in Computer Science*, pages 93–112. Springer-Verlag, Jan. 2000.

[17] R. Levien and A. Aiken. Attack-resistant trust metrics for public key certification. In *Proceedings of the 7th USENIX Security Symposium, San Antonio, TX*, pages 229–242, Jan. 1998.

[18] N. Li, B. N. Grosof, and J. Feigenbaum. A practically implementable and tractable delegation logic. In *Proceedings of the 2000 IEEE Symposium on Security and Privacy*, pages 27–42. IEEE Computer Society Press, May 2000.

[19] S. Marti and H. Garcia-Molina. Limited reputation sharing in p2p systems.

[20] S. Marti, T. Giuli, K. Lai, and M. Baker. Mitigating routing misbehavior in mobile ad-hoc networks. In *Proceedings of MOBICOM 2000*, pages 255–265, 2000.

[21] U. Maurer. Modelling a public-key infrastructure. In E. Bertino, editor, *Proc. 1996 European Symposium on Research in Computer Security (ESORICS' 96)*, volume 1146 of *Lecture Notes in Computer Science*, pages 325–350. Springer-Verlag, 1996.

[22] M. Mohri. Semiring frameworks and algorithms for shortest-distance problems. *J. Autom. Lang. Comb.*, 7(3):321–350, 2002.

[23] M. K. Reiter and S. G. Stubblebine. Resilient authentication using path independence. *IEEE Trans. Comput.*, 47(12):1351–1362, December 1998.

[24] M. K. Reiter and S. G. Stubblebine. Authentication metric analysis and design. *ACM Trans. Inf. Syst. Secur.*, 2(2):138–158, May 1999.

[25] G. Rote. Path problems in graphs. *Computing Supplementum*, 7:155–189, 1990.

[26] D. Watts and S. Strogatz. Collective dynamics of "smallworld" networks. *Nature*, 393, 1998.

[27] W. H. Winsborough, K. E. Seamons, and V. E. Jones. Automated trust negotiation. In *DARPA Information Survivability Conference and Exposition*, January 2000.

[28] P. R. Zimmermann. *The Official PGP User's Guide.* MIT Press, 1995.