A Framework for Dual-Agent MANET Routing Protocols

Brian L. Gaines and Mahalingam Ramkumar
Department of Computer Science and Engineering
Mississippi State University, MS.

***Abstract*— Nodes forming mobile ad hoc networks (MANET) nodes can be logically partitioned into 1) a selfish user agent serving the interests of the end user (owner) of the mobile device and 2) a selfless network agent serving the broad interests of the network. The routing tasks performed by any mobile node can then be shared between physically distinct user and network agents. The network agent can be a secure co-processor in the form of a SIM card. In the interests of protecting the integrity of the network agent and keeping its cost low, it is desirable to reduce its complexity. We identify four broad classes of attacks that can be perpetrated by the user agents on resource constrained network agents, and propose low complexity strategies within the scope of network agents, to overcome such attacks and ensure the integrity of the routing protocol.**

I. Introduction

A distinguishing feature of mobile ad hoc networks (MANET) is their reduced dependence on infrastructure. For this reason MANET devices have to perform some additional tasks, like routing packets between nodes, that are traditionally performed by the network operators in infra-structured networks. MANET devices are expected to adhere to a set of rules, or *routing protocols*, to cooperatively discover optimal paths and route packets between nodes.

The presence of nodes that do not strictly adhere to the rules can have a deleterious effect on the overall performance of the MANET. *Secure* routing protocols strive to ensure the reliability of the process (to the extent possible) even under the presence of nodes that wilfully propagate misleading information.

A. Dual Agents

The dichotomy of participants as user and network agents is indeed logical for any network. In infra-structured networks the user agents are network hosts (clients and servers) and the network agents are routers. While there is no reason for hosts to *rely* on other hosts, they rely on routers for *delivering* their packets. The network agents (routers) are generally operated by large network providers, and are implicitly trusted due to the benefit of organizational backing. In addition, there is also a clear *physical* separation between the user and network agents.

In MANETs most mobile nodes will simultaneously act as network hosts and routers. A mobile node A can thus be *logically* partitioned into a user agent A_U and a network agent A_N. The tasks performed by the network agent A_N are "self-less" tasks necessary for maintaining the network. The tasks performed by the user agent A_U are selfish tasks, towards satisfying the requirements of the owner of the mobile. While the end-user has vested interest in protecting the health of the user agent, he / she can even stand to benefit by circumventing the tasks to be performed by the network agent. It is thus desirable to enforce a *physical* separation between the two agents. For example, the network agent can carry out its tasks within a trusted tamper-responsive boundary.

In practical deployments the network agent can be a subscriber identity module (SIM card) provided by the network operators to the subscribers of the network. The user agent can be any general purpose computing / communication device (like a mobile phone) with the capability to interact with the SIM card.

1) Constraints on Network Agents: Any routing protocol will require participants to *assimilate* topology information and *advertise* packets with topology information that is (hopefully) consistent with the information assimilated. In dual agent protocols the task of assimilating and advertising routing information is shared between an untrusted user agent and a trusted network agent.

Some important requirements for improving the trustworthiness of network agents are a) low complexity, to enable inexpensive verification of compliance; and b) low power consumption / heat dissipation to facilitate unconstrained shielding strategies. Thus it is very much desirable to ensure that the tasks performed by the network agents demand *low* complexity.

It is obviously impractical for such low complexity network agents to perform all actions required to ensure physical delivery of packets over the network interface. It is indeed impractical to include analog components like transceivers inside the trusted boundary. Thus all communication interfaces of the network agent will necessarily be out of its control - and under the control of the user agent. More specifically, any exchange between two network agents (say A_N and B_N) will have two "men-in-the-middle" - the user agents (A_U and B_U) of the two mobiles.

Not withstanding the two constraints, the trusted network agents of every mobile are expected to ensure that the untrusted user agents have as little freedom as possible to attack the routing protocol.

B. Contributions of this Paper

While some researchers have proposed secured routing protocols which employ a trusted computing module to enforce the integrity of the protocol, most such efforts have not addressed constraints inherent to practical trustworthy computing modules. The contributions of this paper is a framework for secure dual agent protocols identifying broad classes of attacks that can be perpetrated by user agents against resource limited network agents - 1) fabrication; 2) selective dropping; 3) hiding; and 4) replay attacks. We propose low complexity strategies (well within the capabilities of resource constrained network agents) to address such attacks. The specific strategies proposed include the use of Bloom filters [1] and "next secure" approaches employed for securing the domain name system [2].

1) Organization: In Section II we provide a brief overview of routing protocols and a classification of approaches to secure such protocols. We argue why a dual agent approach is both necessary and practical for securing MANET protocols. We put forth arguments against two prevalent assumptions in the literature: that ① "the use of trustworthy computing modules may be impractical due to their cost" and ② "scalable schemes for ad hoc establishment of pairwise secrets using only symmetric primitives are impractical." We show why the two assumptions are also closely linked. We argue that an obvious emerging trend - the dramatically reducing cost of storage for mobile devices - lays to rest concerns regarding the feasibility of scalable schemes (or negates assumption ②) and in turn contributes to negating assumption ①.

In Section III we provide the generic framework for dual agent routing protocols with a practically unlimited number of eligible participants (mobiles) who may participate in any ad hoc sub-net constituted by such participants. The broad classes of attacks and strategies to address such attacks are described. The discussions are intended to be non specific to the type of ad hoc routing protocols. However, for description of the attacks specific protocols are used as illustrative examples.

2) Notations Used: The following notations are used in this paper. Upper case alphabets like $A, B, C \cdots$ represent unique IDs of (mobile) nodes; the user agent of a mobile node A is represented by A_U. The network agent of A is A_N. The pairwise secret between the network agents of A and B (between A_N and B_N) is K_{AB}.

II. Background

MANET routing protocols can be classified into proactive and reactive protocols. Proactive approaches like destination sequenced distance vector (DSDV) protocol strive to maintain a consistent view of the entire network at all times. In *reactive* protocols like DSR and AODV routes are determined on demand. In reactive protocols the discovery of routes starts with a ROUTE-REQUEST query which is flooded in a controlled fashion, and trigger a ROUTE-RESPONSE from nodes which have knowledge of a route to the destination.

Attacks against routing protocols can be broadly classified as active and passive attacks. The intention of passive attacks is to 1) snoop on private exchanges between end-points or 2) to selfishly conserve resources through non participation in routing / forwarding. Active attacks involve advertising deliberately misleading routing information.

A. Secure Routing Protocols

Secure routing protocols, which strive to address such attacks, can be broadly classified into three categories. In the first category are protocols which rely on cryptographic authentication techniques for verifying the integrity of routing information [3] - [5]. In the second category are schemes which rely on assigning trust metrics to nodes based on first-hand and second-hand observations [6], [7]. In the third category are schemes which rely on trusted computing modules to enforce compliance to routing protocols [8] - [10].

1) Schemes Employing Trusted Computing Modules: In [9] the authors include the wireless transceiver inside the trust boundary. In [8] the trusted computing module has complex features built into the wireless driver (executed within the confines of the trusted module) to verify the integrity of the wireless transceiver. In [10] explicit consideration is given to the need for lowering the complexity of tasks to be performed inside the trusted boundary. The scheme employs "nuglets of currency" protected by smart-cards to promote faithful forwarding of packets.

However, such schemes have not provided adequate consideration for optimal task sharing strategies between trusted components and untrusted components of mobile devices. Arguably, bringing the wireless transceiver within the scope of the protected boundary as in [9] or including complex features in wireless drivers (executed within the trusted boundary) as in [8] implies high cost for practical realization of such trust modules. All such schemes [8] - [10] (including [10] which explicitly strives to reduce overheads within the trusted boundary) assume that the trusted computing modules should be capable of performing asymmetric cryptographic computations, which raises the bar for the capabilities and the cost of such modules. Furthermore, all such efforts have only targeted strategies for securing memory-less on-demand protocols where nodes do *not have to store* routing information.

B. Practicality of the Dual Agent Approach

Physical separation between network and user agents is indeed *existing* practice in mobile telephone networks, where the network agent of a mobile A is a subscriber identity module (SIM card) provided by the network operator. While the use of a network agent in MANET devices assumes the existence of an infrastructure maintained by network operators to distribute SIM cards, this is justifiable, as *pure* ad hoc networks, with absolutely no reliance on any infrastructure (albeit intuitively appealing) are *impractical*.

Firstly, it is well understood that pure ad hoc networks do *not scale* well, as the average bandwidth available to any node in a network of N nodes is proportional to $\frac{1}{\sqrt{N}}$ [11]. Thus practical large scale MANETs will necessarily consist of many ad hoc sub-nets interconnected by wired networks. Secondly, securing MANETs demands measures to

restrict the devices that can take part in co-operative routing. Practical large scale MANET deployments thus have some clear roles for a network service provider like i) maintaining the infrastructure for routing packets *between* ad hoc sub-nets; and ii) *conferring* upon every participant, the eligibility to take part in the network.

1) Practicality of Low Cost Trustworthy Computing Modules: One prevalent assumption in the literature is that "the use of trusted computing modules may not be practical due to cost constraints." A recent work by Ramkumar [12] effectively negates this supposition.

The reason why existing trustworthy computing modules are expensive is that it is implicitly assumed that such modules should be capable of performing asymmetric cryptographic computations. This implicit assumption is a result of another prevalent assumption in the literature - that "scalable schemes for establishment of pairwise secrets are impractical (without mandating asymmetric primitives)."

If asymmetric schemes are not essential, we can substantially reduce the complexity of trusted computing modules. Such a module may possess a single hardware block cipher and an elementary general purpose processor. Low complexity modules can be more readily verified for compliance. A more important implication of modules demanding low power consumption / heat dissipation is that unconstrained shielding strategies can be used to detect and respond to intrusions aimed at modifying the software executed by such modules or exposing secrets protected by the modules [12], [13]. In other words, if we can avoid the need for asymmetric cryptographic operations inside the trusted boundary we can realize trustworthy computing modules at low cost [12].

2) Practicality of Scalable Low Complexity Schemes: In general, scalable schemes for establishing shared secrets (using only symmetric primitives) either require a trusted server for mediation (Kerberos like schemes), or are susceptible to collusions (key predistribution schemes). Schemes that simultaneously overcome the need for a mediator and susceptibility to collusions do not scale well (non scalable key predistribution schemes).

Scalable (and collusion susceptible) KPSs consist of a key distribution center (KDC) who chooses a set of P secrets and an unlimited number of entities with unique IDs. The network size is limited only by the size of the "ID space" as every entity requires a unique ID. Every entity is assigned a set of $k \leq P$ secrets based on their ID to enable any two entities (say A and B) to independently compute a pairwise secret (K_{AB}).

An attacker who has exposed secrets from v nodes $\{O_1 \cdots O_v\}$ may be able to discover K_{AB} even without access to the secrets of A or B. An *n-secure* KPS can "resist" an attacker who has pooled together all secrets of n entities, *irrespective* of the network size N.

A recent trend that dramatically improves the practical appeal of key predistribution primitives is the rapidly reducing cost of storage for almost every conceivable application scenario. Some novel scalable key predistribution schemes have been developed recently to explicitly take advantage of inexpensive storage resources to realize very high levels of collusion resistance - high enough to render their "susceptibility to collusions" irrelevant in practice.

For example, the scheme in [12] requires about 48 MB of storage for every node to resist collusions of $n = 100,000$ nodes pooling their secrets together, irrespective of the network size N. The storage does not need to be inside the trusted boundary. The computational overheads for computing a pairwise secret amounts to a few tens of hash function evaluations. This modest computational requirements render them very well suited for low complexity modules.

III. Dual Agent Routing Protocols

In the dual agent approach every mobile device employs a resource limited and tamper-proof SIM card as a network agent. The network consists of the network operator and an *unlimited* number N of subscribers. Subscribers can be inducted into the network at any time by the network operator by providing the subscriber with a SIM card with a unique ID and a unique secret. The subscribers may employ any mobile device, manufactured by any vendor, as long as it can accept and "interact" with the SIM card. The unique ID (say 64-bit ID to cater for "practically unlimited" N) of the SIM card also serves as the address of the mobile device. The SIM card, with autonomous processing capabilities, is the network agent. All other components of the mobile device constitute the user agent. The mobile device is assumed to be comparable to a modern mobile phone (in capability).

Any inducted mobile device (with a SIM card) can take part in any ad hoc sub-net created by (in general) a subset N' of the N inducted devices and can join / leave any ad hoc sub-net at any time. For example, a small subset of the N devices may find themselves in some location like an airport or a mall or spread over a sparsely inhabited rural area, and form an ad hoc sub-net.

A. Key Distribution for Pairwise Secrets

The key distribution scheme permits any of the N mobile devices to establish a shared secret with any mobile device inducted into to the network. A key distribution center (managed by the network operator) provides every participant with secrets. The secrets assigned to a mobile can be encrypted using the unique secret of the mobiles SIM card, and distributed over any network (public FTP sites or even through optical storage disks over postal network). The secrets assigned to the network agent A_N of mobile device A facilitates A_N to discover a shared secret K_{AB} with the network agent B_N of B. All secrets assigned to A_N are *stored* by the user agent A_U. However the secrets are stored encrypted using a secret privy only to the network agent (stored inside the SIM card).

The user agents are never provided with clear access to the network agent secrets. All computations that employ network agent secrets are performed inside the protected boundary of the SIM card. For this reason we wish to limit network agents to purely symmetric cryptographic operations. In such a case,

a single hardware block cipher in the SIM card can be reused for all computations.

If the scheme in [12] is used for establishing pairwise secrets, computing a pairwise secret K_{AB} by A will require the user agent of A (or A_U) to provide the SIM card (network agent A_N) with 24 encrypted secrets (from the set of perhaps millions of encrypted secrets stored by the user agent). The operations performed inside the SIM card involve 24 block cipher operations and 24 hash function evaluations[1]. The user agent will require to reserve 48 MB in its flash card for the encrypted secrets. As mentioned earlier, such a scheme can resist an attacker who has exposed secrets of 100,000 network agents.

B. Task Sharing for Co-operative Routing

Some of the routing related tasks to be performed by a MANET node include 1) maintaining one-hop and perhaps sub-net wide topology information, 2) advertising topology information that is consistent with the stored information 3) making routing decisions based on topology information and 4) performing cryptographic computations for authentication / verification of routing information.

The routing tasks are shared between the user and network agents. The purpose of task sharing is to facilitate the trusted network agents to ensure that end users (who are assumed to have complete control over the operation of the user agents) have as little freedom as possible to attack the co-operative routing protocol.

The relationship between the network agent (SIM card) and the user agent is not unlike that between an Executive and a Secretary. Any packet leaving a mobile A is "prepared" by the Secretary (user agent A_U - or more specifically the more capable computer in the mobile phone) and submitted to the Executive (network agent A_N - the SIM card) for authentication. Such a routing packet may be received by the user agent B_U of B who simply submits the packet to the network agent B_N for verification of the authentication appended. Verified packets are then handed back to the user agent B_U for storage / further processing.

The Secretary (user agent) thus collates and stores all routing information. The Executive (network agent) simply authenticates the packets. However the user agent has to *convince* the network agent that the information to be authenticated is *consistent*, and thus *deserves* to be authenticated.

1) Routing Records: Any routing packet advertised by a participant may consist of one or more elementary pieces of routing information (or routing *records*). For example, a table of distance vector updates $\mathbf{D_A}$ from a node A indicates the distance from A to many other nodes in the network. Each row of $\mathbf{D_A}$ (say d^i_A) is a record which indicates the distance to some specific node (from A). In general, a routing record can consist of a subject, some routing information, time of creation (of the record) and / or end-of-life (or duration of validity) of the information.

[1]Hash functions can also defined to use the block cipher [15].

The update $\mathbf{D_A}$ is authenticated by A_N before it is transmitted by the user agent A_U. The user agent of a node B (say a neighbor of A) supplies $\mathbf{D_A}$ to its network agent B_N for verification. After verifying the authentication appended by A_N, the network agent B_N 1) parses $\mathbf{D_A}$ into individual records, 2) individually authenticates each record d^i_A by appending a HMAC based on a secret privy only to B_N and 3) hands over such records to the user agent B_U for storage.

2) Spontaneous and Non-spontaneous Packets: Routing packets can be classified into 1) independent or *spontaneous* packets, and 2) dependent or *non spontaneous* packets. Examples of spontaneous packets include ALIVE messages sent periodically by every node to all nodes within range and ROUTE-REQUEST packets in on-demand protocols. The contents of such spontaneous packets do *not* depend on other routing packets. On the other hand, the contents of non spontaneous packets depend on *other* spontaneous and non-spontaneous packets.

For example, in link-state based routing protocols the link-state information supplied by a node A is based on the ALIVE packets received by node A from its neighbors. Similarly in distance vector based protocols when a node A propagates information about the shortest distance to some node B, this information is based on the claimed distance to B by the neighbors of the node A. For authenticating non-spontaneous packets the user agent will have to supply some proof (for example, stored authenticated information obtained from other spontaneous or non spontaneous packets) to the network agent. However the user agent of any node does *not* have to "convince" the network agent to authenticate a spontaneous packet.

3) Attacks on Resource Limited Network Agents: Depending on the extent of storage possible inside the trusted confines of the network agent, the user agent may have different extents of freedom for perpetrating misrepresentation of routing information. Such attacks can be broadly classified into 1) fabrication 2) selective dropping 3) omission and 4) replay (of records) attacks.

Fabrication is the intentional creation or modification of data by a user agent before providing it to the network agent.

As the user agent controls all communication interfaces, the user agent A_U has the ability to drop packets meant for the network agent A_N. Similarly, the user agent A_U may not transmit some routing packet (authenticated by its network agent A_N) and entrusted to A_U for delivery. While indiscriminate dropping of packets by the user agent may be more challenging to address, it is desirable to enforce some measures to ensure that selective dropping is not feasible. It is also desirable to ensure that the user agent either benefits by not dropping packets or alternately, stands a risk of being penalized for dropping packets.

Omission is the intentional act of withholding existing data from the network agent. Due to the limited storage capabilities of the network agent, most of routing related data must be kept externally - or stored by the user agent. In such a scenario if A receives a request for a route to a node B, the user agent

could simply claim that no route for B exists in its routing tables (while a route actually exists). Ideally the network agent A_N should be able to *verify* the assertion by A_U (that a route to B does not indeed exist).

Any routing record (say a record stored by A providing information about the path to B) is authenticated by the network agent A_N before it is handed over to its user agent for storage. Consider a scenario where a routing record has been authenticated indicating a validity period until time T. However, due to changes in topology A_N may receive an updated information at some time $T' < T$ invalidating the older record. Under such conditions the network agent should be able to "remember," during the duration between T' and T, that the authenticated information is no longer valid. If this is not feasible for storage limited network agents, the user agent may be able to *replay* the authenticated record (indicating validity till time T) for the duration between T' and T.

Fabrication and replay attacks are *active* attacks, as they lead to advertisements of deliberately misleading routing information. On the other hand, dropping (selective or indiscriminate) and omitting routing records are passive attacks as they lead to non-participation of the node in the routing protocol.

C. Fabrication

The network agent A_N can either receive fresh routing data either from other network agents or stored routing information from its user agent A_U. The former, (say a packet received from B) can be cryptographically authenticated using a secret shared (say K_{AB}) between the network agents to ensure that the user agents A_U or B_U cannot modify the packet. The latter, for example a routing record indicating a path to B is self-authenticated by the network agent A_N, using a secret privy only to A_N, before providing it to A_U for storage. Thus as long every network agent shares a secret with the network agents of all its neighbors, and if all network agents are trustworthy, fabrication attacks can be easily dealt with.

D. Selective Dropping

One strategy to limit the ability to selectively drop packets is to encrypt the packets with a secret privy only to the network agents. The user agents can gain access to the contents of the packet only *after* they submit the packet to their network agent for decryption and verification. While the user agent can still drop packets, he / she may not be able to selectively drop packets to engineer sophisticated attacks.

The effectiveness of encryption as a strategy to address selective dropping attacks will vary with the nature of the routing protocol. For example, such an approach can be more effective in table-based protocols as many records are grouped together in the routing table, and the user agent will have to drop entire updates from neighbors. However this is not the case for on-demand protocols or even link-state approaches.

1) Logical Neighborhood: The network agent A_N, say with neighbors B, C, D can convey a secret K_A to the network agents of its neighbors. All packets sent from A may be encrypted using the secret K_A. More generally, the network agent A_N may provide the secret only to a subset of its neighbors. For example, the network agent can provide K_A only to D and B, thereby cutting off C from its *logical* neighborhood.

In response to any packet broadcast by A (provided by A_N to A_U for broadcast) to all its neighbors A_N will expect an acknowledgment from the network agents of all nodes in its logical neighborhood. The network agent can also maintain a count of unacknowledged packets (for every neighbor). If the percentage of unacknowledged packets from a neighbor (say C) is below a certain threshold, A_N can cut off C from its logical neighborhood. If the acknowledgments from all neighbors are missing for some packets A_N can conclude that A_U drops packets and stop A from taking further part in the network. This simple strategy of maintain a proactive logical neighborhood can serve as an effective deterrent for user agents that drop packets (both indiscriminate and selective dropping).

E. Omission Attacks

As the resource constrained network agent may have very little memory, it may not be able to store all relevant history of routing records. Consider a scenario where the network agent had received (in the past) a record containing routing information for node 10. In response to a query for a route to a node 10 the user agent (who **does** have the route to 10 in its cache) may simply claim that no record for 10 exists. To overcome such attacks the user agents should be able to provide *authenticated denial* of existence of routing records.

Such a strategy is used in DNSSEC [2] in the form of NSEC records for *authenticated denial of existence* of DNS records. The stored records can be arranged in some canonical order like ascending order of the subject IDs. If at some point in time the user agent's stored records consists of (records corresponding to) subjects $4, 10, 12, \cdots$, the authentication information appended to the record for subject 10 includes the ID of the *next* subject 12. Thus when a query for subject 11 is received, the user agent can offer authenticated denial to the network agent that an entry for 11 does *not* exist (as the record for 12 follows the record for 10).

In DNSSEC the authenticated denial should verifiable by *any* querier. However, for our case the authenticated denial provided by the user agent needs to be verified *only* by its network agent. Thus while digital signatures are required for NSEC in DNSSEC, a hashed message authentication code[2] (HMAC) based on a secret (privy only to the network agent) is sufficient in our case.

However, in DNSSEC the entire set of resource records (in the form of a master file) are provided to the authenticator at the same time. NSEC does not readily cater for dynamic modifications (insertion of new records or deletion of records) to the master file. Unfortunately in our case the routing information records will be received and processed asynchronously

[2]A HMAC for a message M based on a key K is computed as $h(M, K)$ where $h()$ is a cryptographic hash function (like SHA-1).

by nodes and such records may also have different end-of-life (end of validity period) after which they have to be discarded.

The need for asynchronous introduction of records opens up replay attacks.

F. Replay Attacks

Consider a scenario where a user agent with records $4, 10, 12, \cdots$ receives an additional record for subject 6. The authenticated record stating that "4 NEXT 10" is invalidated and replaced by "4 NEXT 6" and "6 NEXT 10." However, when a query for 6 is received at a later time the user agent may be able to replay the invalidated record ("4 NEXT 10") for "authenticated denial" of record 6 (while it obviously exists).

The problem of rendering a previously validated information indicating end-of-life at time T, invalid a some time $T' < T$ is similar to the problem of revoking a public key certificate (indicating validity till time T but revoked at a time $T' < T$). The list of revoked certificates (or superseded routing records in our case) have to be maintained. For this purpose the network agents have to maintain a concise summary (for example a hash of a revoked record) for a duration between T' and T (during which time replay attacks are feasible).

1) Bloom Filters: Bloom filters [1], which are randomized fixed-size data structures for inserting and querying items belonging to a set, are useful tools for this purpose [14]. A (k, t) Bloom filter employing t hash functions (which produce outputs in the range $0 \cdots k-1$) is initialized as a string of $k >> t$ zero-bits. When a record R is "added" to the Bloom filter, t of the k bits of the Bloom filter are set to 1. Typically many such records are added to the Bloom filter, and each sets some bits.

Given some record R' the Bloom filter permits one to decide if R' was previously added to the Bloom filter. If it is observed that at least one bit corresponding to the record R' is *not* set, we can safely conclude that R' was *not* added to the Bloom filter earlier. However, if all the bits corresponding to R' have been set in the Bloom filter, we *cannot* conclude with certainty that R' was added to the Bloom filter earlier. It is possible that the bits set by a set of records $R_1, R_2, \cdots R_n$ added to the Bloom filter had set all the bits corresponding to the record R', leading to a false positive.

For purposes of maintaining a list of revoked routing records (which have been superseded by newer records) the implication of false positives is that a non-revoked record R supplied by the user agent (to the network agent) can be misinterpreted by the network agent as revoked. However, if the network agent had ensured that every record added to the Bloom filter had set *at least* one new bit, it is possible for the user agent to demonstrate to the network agent as to *why* the false positive occurred. The user agent can simply provide the list of records previously added to the Bloom filter which (together) set all the t bits set by R. Thus, the issue of false positives in Bloom filters is not a limiting factor for the intended application. More specifically, the overheads for dealing with false positives is borne by the user agent - not the network agent.

As all the computations necessary for Bloom filters can be carried out by a block cipher (a hash function can be implemented using a block cipher [15]), such strategies are well within the scope of the network agent.

IV. Conclusions

We proposed a framework for dual agent secure routing protocols where every mobile device consists of an untrusted user agent and a trustworthy network agent with modest capabilities. The network agents are constrained to perform only symmetric cryptographic computations and efficiently reuse a single hardware block cipher. The network agent strives to keep the user agent (all other components of the mobile device) in check to ensure adherence to the rules to be followed for co-operative routing. We provided a taxonomy of attacks and enumerated that low complexity strategies well within the scope of network agents to address such attacks.

Application of broad strategies proposed in this paper to specific ad hoc routing protocols are being addressed currently by the authors.

References

[1] B. Bloom, "Space / Time Tradeoffs in Hash Coding with Allowable Errors," Communications of the ACM, 13(7):422-426, July 1970.

[2] R. Arends, R. Austein, M. Larson, D. Massey, S. Rose "RFC 4033: DNS Security Introduction and Requirements," March 2005.

[3] P Papadimitratos, Z. J.Haas, "Secure Routing for Mobile Ad Hoc Networks," Proceedings of the SCS Communication Networks and Distributed Systems Modeling and Simulation Conference(CNDS 2002), San Antonio, Texas ,2002

[4] Y-C Hu ,A Perrig,. D B.Johnson, "Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks," Journal of Wireless Networks, **11**, pp 11–28, 2005.

[5] M G Zapata, N.Asokan ,"Securing Ad Hoc Routing Protocols," Proceedings of the ACM workshop on Wireless security, Atlanta, Georgia, September 2002.

[6] J.Marshall, V.Thakur, A.Yasinsac,"Identifying flaws in the secure routing protocol," Proceedings of the 2003 IEEE International Performance, Computing, and Communications Conference, 2003.

[7] A. A. Pirzada, C. McDonald, "Trusted Route Discovery with TORA Protocol," Proceedings of the Second Annual Conference on Networks and Services Research (CNSR), 2004.

[8] M. Jarrett and P. Ward, "Trusted Computing for Protecting Ad-hoc Routing," Proceedings of the 4th Annual Communication Networks and Services Research Conference, IEEE Computer Society, May 2006.

[9] J-H. Song, V. Wong, V. Leung, "Secure Routing with Tamper Resistant Module for Mobile Ad Hoc Networks," ACM SIGMOBILE Mobile Computing and Communications Review, vol. 7, no. 3, ACM Press, New York, Jul. 2003.

[10] J-P. Hubaux, L Buttyan, S. Capkun, "Quest for Security in Mobile Ad Hoc Networks," Proceedings of the ACM MOBIHOC 2001.

[11] P. Gupta, P. R. Kumar, "The capacity of wireless networks," IEEE Trans. Inform. Theory, vol. 46, no. 2, pp. 388404, Mar. 2000.

[12] M. Ramkumar, "Trustworthy Computing Under Resource Constraints With the DOWN Policy," IEEE Transactions on Secure and Dependable Computing, to appear.

[13] S. W. Smith, *Trusted Computing Platforms: Design and Applications,* Springer, New York, 2005.

[14] A. Broder, M. Mitzenmacher, "Network Applications of Bloom Filters: A Survey," *Internet Mathematics*, vol. 1, no. 4, pp 485–509, 2005.

[15] A. J. Menezes, P. C. van Oorschot, S. A. Vanstone. *Handbook of Applied Cryptography*, August 2001.