Report on a Working Session on Security in Wireless Ad Hoc Networks

Levente Buttyán
levente.buttyan@epfl.ch

Jean-Pierre Hubaux
jean-pierre.hubaux@epfl.ch

Laboratory for Computer Communications and Applications
Swiss Federal Institute of Technology – Lausanne (EPFL), Switzerland

On June 12, 2002, we organized a working session devoted to the topic of security in wireless ad hoc networks. This event took place on our campus the day after MobiHoc 2002 and attracted around twenty persons in an informal setting.

Securing wireless ad hoc networks is particularly difficult for many reasons including the following:

- *Vulnerability of channels.* As in any wireless network, messages can be eavesdropped and fake messages can be injected into the network without the difficulty of having physical access to network components.

- *Vulnerability of nodes.* Since the network nodes usually do not reside in physically protected places, such as locked rooms, they can more easily be captured and fall under the control of an attacker.

- *Absence of infrastructure.* Ad hoc networks are supposed to operate independently of any fixed infrastructure. This makes the classical security solutions based on certification authorities and on-line servers inapplicable.

- *Dynamically changing topology.* In mobile ad hoc networks, the permanent changes of topology require sophisticated routing protocols, the security of which is an additional challenge. A particular difficulty is that incorrect routing information can be generated by compromised nodes or as a result of some topology changes, and it is hard to distinguish between the two cases.

Clearly the problem is so broad that there is no way to devise a general solution. It is also clear that different applications will have different security requirements. The complexity and diversity of the field has led to a multitude of proposals, which focus on different parts of the problem domain. The presentations of the working session reflected this complexity and diversity.

The working session was started with a brief overview given by J.-P. Hubaux on the different aspects of security in wireless ad hoc networks. The remaining presentations were organized into the following four sessions:

- *Trust and key management.* Many security objectives can be achieved by using cryptographic mechanisms. Cryptographic mechanisms, in turn, rely on the proper management of cryptographic keys. The presentations by L. Zhou, S. Lu, and S. Čapkun were strongly related to this problem, and in particular, to certificate based public-key distribution in mobile ad hoc networks. The talk given by G. Tsudik addressed the broader issue of membership management in dynamic peer groups, and went beyond the problems of group key management.

- *Secure routing and intrusion detection.* Existing ad hoc routing protocols, such as DSR and AODV, are vulnerable to many kinds of attacks. It is fairly easy to inject fake routing messages or modify legitimate ones such that the operation of the network would be heavily disturbed (e.g., by creating loops or disconnecting the network). The talks given by Z. Haas, A. Perrig, Y.-C. Hu, and E. Belding-Royer addressed this problem by proposing secure ad hoc routing protocols that are resistant to various kinds of attacks. In his presentation, C. Castelluccia suggested the use of crypto based identifiers for securing ad hoc routing protocols. Finally, the talk by Y. Zhang focused on the problem of intrusion detection in ad hoc networks.

- *Availability.* This session was concerned with the problem of service unavailability due to either intentional denial of service attacks or selfishness of the nodes. Selfishness is a new problem that arises specifically in the context of ad hoc networks where the nodes belong to multiple administrative domains. In these networks, nodes may tend to deny providing services for the benefit of other nodes in order to save their own resources (e.g., battery power). The presentation by N. Vaidya discussed the problem of greediness (a form of selfishness) at the MAC layer, while R. Molva, S. Buchegger, and L. Buttyán addressed selfishness in the context of packet forwarding.

- *Cryptographic protocols.* Traditional solutions for key management can be unsuitable for ad hoc networks; likewise, existing solutions for other, higher level security services, may also have to be reconsidered. An example is fair exchange, which is known to be impossible without a trusted third party, hence, its implementation can be problematic in an infrastructureless ad hoc network. The presentations by S. Vaudenay and L. Buttyán addressed this problem by proposing concepts that provide weaker guarantees than true fairness but can be implemented in ad hoc networks.

What follows is a set of extended abstracts of the presentations. The abstracts have been written by the participants themselves; we only collected them together and did some editorial work. We would like to thank all of the participants for their contribution. We are also grateful

to the Swiss National Competence Center in Research on Mobile Information and Communication Systems (NCCR-MICS)[1], also known as the Terminodes Project[2], for sponsoring the working session. Finally, many thanks to Claude Castelluccia who suggested to publish this report in MC^{2}R.

Trust and key management

Distributed Trust in Ad Hoc Networks, *Lidong Zhou (Microsoft Research, Mountain View CA)*

We propose a security paradigm centering around the notion of *distributed trust* for ad hoc networks. Distributed trust enhances security by composing otherwise untrustworthy individual entities into a trustworthy aggregation, one that remains available and correct even if some of its entities fail or become compromised. The challenge of constructing such a trustworthy aggregation lies not only in how to create and configure the aggregation, but also in how the aggregation maintains its security by adapting to changes in the network topology and the environment, as well as to compromises of the individual entities. We demonstrate how we apply distributed trust to building secure services and to secure routing.

Distributed Secure Services: For a security-sensitive service, such as a certification authority, distributed trust advocates providing the service through a set of nodes as *servers*, so that the service remains available and correct even if a small number of servers become compromised.

Fault tolerance mechanisms, such as the replicated state-machine approach and quorum systems, have proven effective against server failures. However, replication of a secret on servers increases the chance of disclosure because compromise of any server exposes the secret. The solution here is secret sharing. A secret sharing scheme allows servers to store shares of a secret, so that the secret can be recovered if and only if enough shares are obtained from servers. *Threshold cryptography* can be used for servers to perform signing and decryption when the secret is a private key.

Secret sharing alone does not defend against *mobile adversaries*, which attack, compromise, and control one server for a limited period before moving to the next. Over time, a mobile adversary could compromise enough servers and recover the secret. *Share refreshing*, where servers create a new, independent set of shares and replace old shares with the new ones, provides a defense to mobile adversaries. Because new shares cannot be combined with old ones to recover the secret, a mobile adversary must compromise enough servers between two consecutive executions of share refreshing. Share refreshing can be generalized for the new shares to be re-distributed to a different set of servers with a possibly different configuration. Such generalization allows a service to adapt itself when certain servers are permanently compromised or when the service encounters a more malicious environment.

The feasibility of building such a secure distributed service has been demonstrated for networks such as the Internet. The obstacles to deploying such a service on an ad hoc network come not only from the scarcity of both computation and communication resources in an ad hoc network, but also from the lack of secure network infrastructure. The latter problem is addressed by secure routing.

Secure Routing: Secure routing is concerned with maintaining connectivity in an ad hoc network despite compromised nodes disrupting route discovery and message transmission. Following the philosophy of distributed trust, multiple (ideally disjoint) paths between two nodes should be discovered and maintained so that a small number of compromised nodes cannot disrupt all the paths. We believe route discovery should be coupled with message transmission, because, otherwise, even with a secure route discovery protocol that prevents compromised nodes from cheating, a compromised node could cooperate during route discovery, but misbehave during message transmission if the node happens to be on the discovered path.

We thus propose a probabilistic secure routing scheme, where a node maintains, for each possible destination, a probability distribution over all its neighbors. The probability associated with a neighbor reflects the relative likelihood of that neighbor forwarding and eventually delivering a message to the destination. Every message is routed probabilistically at each hop based on the probability distribution for the destination. Multiple paths are thus implicitly maintained through the probability distributions. Message transmission itself provides the feedback for nodes to adjust the probability distributions. For example, an acknowledgment, whose integrity is cryptographically protected (e.g., by a digital signature), of the receipt of a message provides a positive feedback for the path through which the message traversed. Consequently, the set of multiple paths maintained by the routing scheme for any two nodes adapts itself as the probability distributions change.

Although probabilistic routing schemes have been studied in swarm intelligence, the application to secure routing raises various new issues, such as how to ensure the integrity of feedbacks in face of compromised nodes without excessive costs. Both analysis of and experiments with such schemes are also needed to establish their practicality.

An extensive survey on swarm intelligence can be found in [3]. Other related references can be found in the bibliography of [37, 38].

Acknowledgments: This paper is in part based on joint work with Fred B. Schneider, Robbert van Renesse, and Zygmunt J. Haas, and benefited from discussions with Michael Marsh, Panagiotis Papadimitratos, and Martin Roth.

Network Performance Centric Security Design in MANET, *Hao Yang (UCLA), Gary Zhong (UCLA), and Songwu Lu (UCLA)*

> “In theory there is no difference between theory and practice. In practice there is...”
>
> –Bruce Schneier in *Secrets and Lies* [33]

Security is a basic requirement for mobile ad hoc networks. Several recent papers [37, 16, 19] have started to address

[1] NCCR-MICS is supported by the Swiss National Science Foundation under grant number 5005-67322.

[2] http://www.terminodes.org/

security issues in such networks. While these early proposals each have their own merit, they mainly focus on the security vigor of the design and leave the *network performance* aspect largely unaddressed. As a result, these solutions may be extremely secure from the cryptographic standpoint, but their real performance when deployed in the network is unclear. This concern is further aggravated by the unique characteristics of ad hoc networks, such as highly dynamic network topology, frequent node arrival/departure, and bandwidth-constrained wireless links.

In this work, we shift our main attention from the cryptography-centric design approach to a more network-centric design scheme, and focus on the practical network performance aspect of the security design. Our goal is to develop *network performance-centric* security solutions that effectively balance security strength and network performance in practice.

We focus on node authentication, the basic component in a security solution. At the first stage, we investigate several design choices – centralized, peer-to-peer, and localized authentication schemes, and examine their network performance by extensive simulations. The centralized scheme [37] is similar to the TTP (Trusted Third Party) authentication widely used in the wired networks, in which authentication is done via the third-party certificate authority (CA). The peer-to-peer authentication scheme [16] bears the same philosophy as PGP, where authentication is done through a chain of trust relationship that forms the "web of trust". The localized scheme [19] is specially designed for ad hoc networks, in which each node is authenticated and monitored by its multiple local neighboring nodes.

The simulation results are summarized in Table 1.

Scalability: We examine whether the design scales to the number of nodes. The average node speed is 10 m/s, and there is no channel error and other ongoing traffic (benchmark setting). When the number of nodes increases from 40 to 100, the success ratio of the authentication request in centralized scheme drops from 92% to 22%; the success ratio in peer-to-peer scheme remains stable around 88%; while the success ratio in localized scheme remains stable around 96%.

Availability: We increase the network traffic load and examine whether the design provides "anytime, anywhere" security service to the mobile hosts. For a 60-node setting with average speed of 10m/s, when the network traffic load increases from 0 to 100 pkt/s (packet size 512B), the success ratio in centralized schemes drops from 80% to 45%; the success ratio in peer-to-peer scheme almost remains stable around 85%; while the success ratio in localized scheme remains stable around 95%.

Robustness: We examine the robustness feature for different channel conditions. For the same 60-node setting, when the channel error rate increases from 0 to 10%, the success ratio in centralized scheme drops from 80% to 50%; the success ratio in peer-to-peer scheme drops from 85% to 82%; while the success ratio in localized scheme drops from 95% to 93%.

The fundamental reason for the performance difference is the traffic pattern in these schemes. The localized scheme has the best network performance in that the traffic is not only distributed in the network, but also confined in the local neighborhood. As a result, the impact of network scale, traffic load, channel error, mobility, etc., on the localized authentication service is very small in most scenarios.

The current study provides two guidelines for future security design in ad hoc networks: 1) the network performance aspect should be explicitly considered in the design; 2) in order to have good network performance, it is desirable for the security solution to have localized traffic pattern. Our next-stage effort focuses on devising new network mechanisms to improve the performance of the security design.

Scheme	Centralized	Peer-to-Peer	Localized
Scalability	Bad	Good	Good
Availability	Bad	Uncertain	Good
Robustness	Bad	Uncertain	Good
Communication	Centralized	Distributed	Localized
Computation	Undertaken solely by the servers	Shared by the nodes	Shared by the nodes

Table 1: Network Performance Comparison of Three Authentication Schemes (Network Performance Centric Security Design in MANET, *H. Yang, G. Zhong, and S. Lu*)

Self-Organized Public Key Management for Mobile Ad Hoc Networks, *Srdjan Čapkun (EPFL), Levente Buttyán (EPFL), and Jean-Pierre Hubaux (EPFL)*

By definition, mobile ad hoc networks do not rely on any fixed infrastructure; instead, all networking functions (e.g., routing, mobility management, etc.) are performed by the nodes themselves in a self-organizing manner. In security terms, we consider an ad hoc network to be *fully* self-organized, meaning that there is no infrastructure (hence no PKI), no central authority, no centralized trusted third party, no central server, no secret share dealer, *even in the initialization phase*.

In [16], we propose a self-organizing public-key management system for fully self-organized mobile ad hoc networks. Our approach is similar to PGP in the sense that users issue certificates for each other based on their personal acquaintances. However, in the proposed system, certificates are stored and distributed by the users themselves, unlike in PGP, where this task is performed by online servers (called certificate directories). In the proposed self-organizing public-key management system, each user maintains a *local certificate repository*. When two users want to verify the public keys of each other, they merge their local certificate repositories and try to find appropriate certificate chains within the merged repository that make the verification possible.

The success of this approach very much depends on the construction of the local certificate repositories and on the characteristics of the certificate graphs. By a certificate graph, we mean a graph whose vertices represent public-keys of the users and the edges represent public-key certificates issued by the users. In the same article, we propose several repository construction algorithms and study their performance. The proposed algorithms take into account the characteristics of the certificate graphs in a sense that the choice of the certificates that are stored by each user depends on the connectivity of the user and her certificate graph neighbors. More precisely, each user stores in her local repository several directed and mutually disjoint paths of certificates. Each path begins at the user herself, and each certificate on the path is chosen from the set of certificates that are connected to the last selected user on the path in such a way that the chosen certificate leads to a user that has the highest number of certificates connected to her (i.e., the highest vertex degree). We call this algorithm the *Maximum Degree Algorithm*, as the local repository construction criterion is the degree of the vertices in the certificate graph. In a second, more sophisticated algorithm, certificates are selected into the local repositories based on the number of the *shortcut certificates* connected to the users. Here, a shortcut certificate is defined as a certificate such that when it is removed from the graph, the shortest path between the two users previously connected by this certificate becomes strictly larger than two. We call this algorithm the *Shortcut Hunter Algorithm*.

The analysis of these two algorithms shows that even a simple construction algorithm can achieve high performance in the sense that any user u can find at least one certificate chain to any other user v in the merged repository of u and v with very high probability even if the size of the local repositories is small (in the order of $\sqrt{n}$) compared to the total number n of users in the system. We simulated the effectiveness of the two proposed algorithms on the PGP certificate graph, as this graph is the only known example of a self-organized certificate graph creation. The results show that even with a certificate repository size of less than $\sqrt{n}$, two users have a high (90%) chance of finding an appropriate chain of certificates between them in their merged repositories. Our results further show that the average length of the certificate chains in the merged local repositories of the users is ≈ 8, while in the whole PGP certificate graph it is ≈ 6. This is due to the characteristics of the PGP certificate graph, which exhibits the small-world phenomenon [8]. More precisely, in PGP certificate graphs, the average shortest path length between two vertices is small (compared to the graph size) and it scales logarithmically with the graph size.

As any approach that uses certificate chains, this approach assumes that trust is transitive (which is often not the case in practice). In order to alleviate this problem, we propose to look for multiple certificate paths and to use authentication metrics.

The main idea of the proposed approach is summarized on Figure 1. The public keys of the users are represented by certificate graph vertices, while the graph edges represent public-key certificates issued by the owners of the public keys. The figure shows the local certificate repositories of users u and v and the chains of certificates that u uses to authenticate the public key K_v of v.

Admission Control in Collaborative Groups, *Yongdae Kim (University of Minnesota, Minneapolis), Daniele Mazzocchi (Torino Polytechnic), and Gene Tsudik (UC Irvine)*

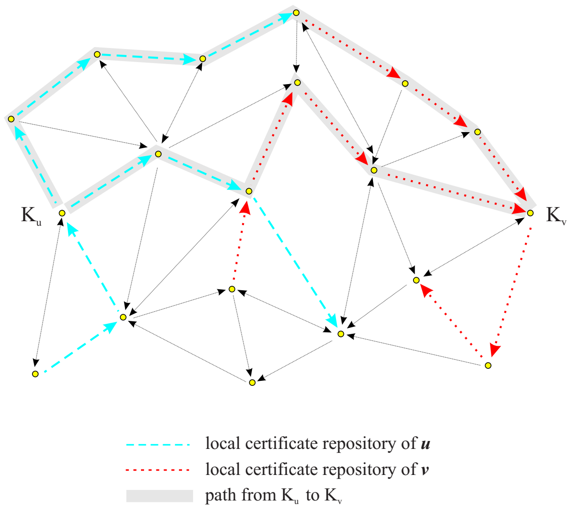
K_u
K_v
local certificate repository of *u*
local certificate repository of *v*
path from K_u to K_v

Figure 1: Paths from K_u to K_v in the merged local repositories of u and v (Self-Organized Public Key Management for Mobile Ad Hoc Networks, *S. Čapkun, L. Buttyán, and J.-P. Hubaux*)

The current proliferation of group-oriented applications, protocols and services triggers the need for specialized group security services and mechanisms. Examples of popular group-oriented settings include: IP telephony, video/audio conferencing, file sharing, collaborative workspaces, and multi-user games. Group settings are clearly very diverse. Some, such as conferencing, require synchronous operation while others, such as file sharing, operate in a disconnected, asynchronous manner. Communication models vary as well: from the one-to-many or few-to-many (e.g., GPS) to any-to-any peer groups (e.g., Gnutella).

The need for, and the importance of, group security mechanisms has been recognized by the research community as supported by popularity of this topic. However, the bulk of prior work has been done in the context of large multicast-style groups where it is natural to assume or impose a centralized authority (the sender or an on-line trusted third party) that can perform security chores, e.g., key management, admission/access control and member authentication. Such an authority may be group-specific or group-independent and its existence makes it relatively easy to implement security policies and mechanisms. However, due to peer nature, some other group settings exhibit unique properties and requirements.

Our research is focused on admission control mechanisms for peer groups. A *peer* group is characterized by a flat non-hierarchical) structure where all members have identical rights and duties. In other words, there is no underlying assumption of a centralized authority that provides security services such as access control or key management. Also, a peer group often involves any-to-any communication: any member can send data to any other member(s). Security in peer groups presents a formidable challenge. Lack of centralized authority entails the involvement of *all* group members in tasks, such as key management. As evidenced by prior work in peer group key management, it is very hard to design multi-party, multi-round protocols that are, at the same time, secure, efficient and robust.

Although an important issue, peer group admission control has been somehow overlooked in the past. With the exception of Antigone, most prior work in peer group security has focused on key management and authentication, whereas, without admission control, key management alone is all but useless. Consequently, our initial goal is to develop a framework for peer group admission control as well as investigate cryptographic mechanisms suitable for different peer group flavors.

This short (1-page) abstract is clearly insufficient to provide the technical description of our work. We refer the reader to the full paper[3] for details. However, as a brief overview, the highlights of the full paper are as follows:

- We begin by motivating the need for a so-called *Group Charter*, a certified statement stipulating the group admission criteria (policy or rules).

- We then argue that an authoritative entity (referred to as the *Group Authority*) must be defined (and reflected in the Group Charter) in order to actually perform the admission. This entity may be off- or on-line and may or may not be distributed. In one extreme case, it is composed of all group members collectively.

- Next, several important dimensions in peer group admission control are considered: Membership Dynamics, Membership Awareness, Members' On-Line Presence, Group Lifetime. Related to these are the characteristics of the Group Authority, in particular, its placement (in or out of the group) and its composition (single or distributed)

- Then, we introduce and discuss three models for peer group admission control: 1) admission via public ACL, 2) admission by Group Authority, and 3) admission by the members themselves (here we also consider the case of the group acting as its own Group Authority)

- Concentrating on the last (and the most challenging) model, we look into different flavors of voting suitable for the admission.

- Lastly, we attempt to sorting out a number of signature schemes applicable for the voting process. These include: 1) plain digital signatures, 2) threshold signatures (both fixed and dynamic), 3) accountable subgroup multi-signatures and 4) group signatures.

To conclude, we motivate the importance of admission control in dynamic peer groups. This work represents an initial attempt to develop an admission control framework suitable for different flavors of peer groups and match them with appropriate cryptographic techniques. We examine various dimensions of admission control, discuss several cryptographic techniques and assess their applicability. Clearly, much remains to be done...

Secure routing and intrusion detection

Secure Routing for Mobile Ad Hoc Networks[4], *Panagiotis Papadimitratos (Cornell University) and Zygmunt J. Haas (Cornell University)*

For such self-organizing infrastructures as mobile ad hoc networks , envisioned to operate in an open, collaborative, and highly volatile environment, the importance of security cannot be underrated. The provision of comprehensive secure communication mandates that both route discovery and data forwarding be safeguarded. The discussed here Secure Routing Protocol (SRP) [25] counters malicious behavior that targets the discovery of topological information. The protection of the data transmission is a separate problem: an intermittently misbehaving attacker could first comply with the route discovery to make itself part of a route, and then corrupt the in-transit data. Protection of data transmission is addressed through our related Secure Message Transmission Protocol (SMT), which provides a flexible, end-to-end secure data forwarding scheme that naturally complement SRP. Here we discuss the design of SRP only, while SMT is the subject of another publication.

SRP provides correct routing information; i.e., factual, up-to-date, and authentic connectivity information regarding a pair of nodes that wish to communicate in a secure manner. The sole requirement is that any two such *end* nodes have a security association. Accordingly, SRP does not require that any of the *intermediate* nodes perform cryptographic operations or have a prior association with the end nodes. As a result, its end-to-end operation allows for efficient cryptographic mechanisms, such as message authentication codes. More importantly, SRP can be used in wide range of networks, without restrictive assumptions on the underlying trust, network size, and membership.

SRP discovers one or more routes whose correctness can be verified from the route "geometry" itself. Route requests propagate verifiably to the sought, trusted destination. Route replies are returned strictly over the reversed route, as accumulated in the route request packet. In order to guarantee this crucially important functionality, the interaction of the protocol with the IP-related functionality is explicitly defined. An intact reply implies that (i) the reported path is the one placed in the reply packet by the destination, and (ii) the corresponding connectivity information is correct, since the reply was relayed along the reverse of the discovered route.

The securing of the route discovery deprives the adversarial nodes of an "effective" means to systematically disrupt the communications of their peers. Despite our minimal trust assumptions, attackers cannot impersonate the destination and redirect data traffic, cannot respond with stale or corrupted routing information, are prevented from broadcasting forged control packets to obstruct the later propagation of legitimate queries, and are unable to influence the topological knowledge of benign nodes. To that

[3] http://sconce.ics.uci.edu/admctl

[4] This work has been sponsored in part by the NSF grant number ANI-9980521 and the ONR contract number N00014-00-1-0564.

extent, SRP provides very strong assurances on the correctness of the link-level connectivity information as well. It precludes adversarial nodes from forming “dumb” relays, and from controlling multiple potential routes per source-destination pair. However, with the adversary within the transmission range of the destination the last two defenses are somewhat weakened. Additionally, two colluding adversaries might be able to “tunnel” the query and the corresponding reply packets to each other within a single query/response phase. Then, the validated route would provide partially correct link information only. However, this vulnerability is not specific to SRP: such information could not be distinguished from the actual link connectivity, even under the assumption of a fully trusted network.

Furthermore, it is important to estimate the cost of introducing security features, such as computational and transmission overhead, increased traffic and delays, etc. On the one hand, security countermeasures should not undermine the efficiency of the network protocols; e.g., the ability of nodes to quickly respond to topological changes and discover correct routes. On the other hand, it necessary to ensure the effectiveness of the security provision; i.e., that the route discovery retains its ability to operate when under attack. Finally, the solution should be applicable to a wide range of network instances, especially when nodes have limited computational and communication resources. Through a systematic performance evaluation, our results show that, over a range of scenarios, SRP is successful in providing correct routing information in a timely manner. Also, it can do so even in the presence of a significant fraction of adversaries that disrupt the route discovery. Moreover, we observe that the processing overhead due to cryptographic operations remains low, allowing the protocol to remain competitive to reactive protocols, which do not incorporate security features at all.

As future work, we intend to investigate complex attacks against SRP and classify them with respect to their impact on the protocol performance. Through combining of SMT with SRP, the detrimental effects on performance of such attacks, in particular, those of intermittently misbehaving nodes, can be alleviated.

Packet Leashes: A Defense against Wormhole Attacks in Wireless Ad Hoc Networks, *Yih-Chun Hu (Rice University), Adrian Perrig (UC Berkeley), David B. Johnson (Rice University)*

We describe the *wormhole attack* [14], a severe attack against ad hoc routing protocols that is particularly challenging to defend against. We show how an attacker can use the wormhole attack to cripple a range of ad hoc network routing protocols. In the wormhole attack, an attacker records packets (or bits) at one location in the network, tunnels them to another location, and retransmits them there into the network. Most existing ad hoc network routing protocols, without some mechanism to defend them against the wormhole attack, would be unable to find routes longer than one or two hops, severely disrupting communication.

If a wormhole attacker tunnels all packets through the wormhole honestly and reliably, no harm is done; the attacker actually provides a useful service in connecting the network more efficiently. However, when an attacker forwards only routing control messages, routing may be severely disrupted. For example, when used against an on-demand routing protocol such as DSR [17] or AODV [29], a powerful application of the wormhole attack can be mounted by tunneling each ROUTE REQUEST packet directly to the destination target node of the REQUEST. This attack prevents routes more than two hops long from being discovered. Periodic protocols are also vulnerable to the same attack. For example, OLSR [31] and TBRPF [1] use HELLO packets for neighbor detection, so if an attacker tunnels to B all HELLO packets transmitted by A, and tunnels to A all HELLO packets transmitted by B, then A and B will believe that they are neighbors, which would cause the routing protocol to fail to find routes when they are not actually neighbors. The wormhole attack is also dangerous in other wireless applications. One example is any wireless access control system that is proximity based, such as wireless car keys, or proximity and token based access control systems for PCs [11, 18]. In such systems, an attacker could relay the authentication exchanges to gain unauthorized access.

Our solution to the wormhole attack is *packet leashes*. We consider specifically two types of packet leashes: *geographical leashes* and *temporal leashes*. The key intuition is that by authenticating either an extremely precise timestamp or location information combined with a loose timestamp, a receiver can determine if the packet has traversed a distance that is unrealistic for the specific network technology used.

Temporal Leashes: Temporal leashes rely on extremely precise time synchronization and extremely precise timestamps in each packet. The travel time of a packet can be approximated as the difference between the receive time and the timestamp. To be more conservative, however, a node may choose to add the maximum time synchronization error, on the assumption that the sender's clock may be faster than the receiver's. Conversely, to allow all direct communication between legitimate nodes, a node may subtract the maximum time synchronization error, on the assumption that the sender's clock may be slower than the receiver's.

Given the precise time synchronization required by temporal leashes, we constructed some very efficient broadcast authenticators based entirely on symmetric primitives. In particular, we extend the TESLA broadcast authentication protocol [30] to allow the disclosure of the authentication key within the packet that is authenticated. We use a Merkle tree [22] to authenticate these keys.

We demonstrated the suitability of temporal leashes with our authentication mechanism by measuring computational power and memory currently available in mobile devices. Current commodity wireless LAN products such as commonly used 802.11b cards provide 11 Mbps at 250 meters. With time synchronization provided by a Trimble Thunderbolt GPS-Disciplined Clock [34], the synchronization error can be as low as 183 ns with probability $1-10^{-10}$. We also assume authentic keys are re-established every day, with a 20 byte minimum packet size and an 80-bit message authentication code length. Our scheme requires just 2.6 megabytes of storage and requires a maximum of 39,500 hash functions per second (assuming packets are arriving at link speed), which is well within the capability of an iPaq, with 82.2% of its CPU time to spare.

Geographical Leashes: Another method to construct a leash is to use location information and loosely synchronized clocks. If the clocks of the sender and receiver are synchronized to within $\pm\Delta$, and ν is an upper bound on the velocity of any node, then the receiver can compute an upper bound on the distance between the sender and itself d_{sr}. Specifically, based on the timestamp t_s in the packet, the local receive time t_r, the maximum relative error in location information δ, and the locations of the receiver p_r and the sender p_s, d_{sr} can be bounded by $d_{sr} \leq ||p_s - p_r|| + 2\nu \cdot (t_r - t_s + \Delta) + \delta$.

In certain circumstances, bounding the distance between the sender and receiver d_{sr} cannot prevent wormhole attacks; for example, when obstacles prevent communication between two nodes that would otherwise be in transmission range, a distance-based scheme would still allow wormholes between the sender and receiver. A network that uses location information as a leash can control even these kinds of wormholes. To accomplish this, each node has a radio propagation model. A receiver verifies that every possible location of the sender (a $\delta + \nu(t_r - t_s + 2\Delta)$ radius around p_s) can reach every possible location of the receiver (a $\delta + \nu(t_r - t_s + 2\Delta)$ radius around p_r).

Summary: Packet leashes restrict an attacker's ability to tunnel a packet between two legitimate nodes. If deployed in a very conservative way, it may also restrict legitimate communications between two legitimate nodes, but more severely restrict tunnels. Conversely, less conservative settings allow more legitimate communications, but also provide an attacker with greater flexibility.

Secure Efficient Distance Vector Routing in Mobile Wireless Ad Hoc Networks, *Yih-Chun Hu (Rice University), David B. Johnson (Rice University), Adrian Perrig (UC Berkeley)*

We describe our protocol, which we call the *Secure Efficient Ad hoc Distance vector* routing protocol (SEAD) [13]. SEAD is robust against multiple uncoordinated attackers creating incorrect routing state in any other node, even in spite of active attackers or compromised nodes in the network. We base the design of SEAD in part on the *Destination-Sequenced Distance-Vector* ad hoc network routing protocol (DSDV) [28]. In order to support use of SEAD with nodes of limited CPU processing capability, and to guard against Denial-of-Service attacks in which an attacker attempts to cause other nodes to consume excess network bandwidth or processing time, we use efficient *one-way hash functions* and do not use asymmetric cryptographic operations in the protocol.

Distance Vector Routing: In distance vector routing, each router maintains a routing table listing all possible destinations within the network. Each entry in a node's routing table contains the address (identity) of some destination, this node's shortest known distance (usually in number of hops) to that destination, and the address of this node's neighbor router that is the first hop on this shortest route to that destination; the distance to the destination is known as the *metric* in that table entry. Each router forwarding a packet uses its own routing table to determine the next hop towards the destination.

To maintain the routing tables, each node periodically broadcasts routing update containing the information from its own routing table. Each node updates its own table using the updates it hears, so that its route for each destination uses as a next hop the neighbor that advertised the smallest metric in its update for that destination; the node sets the metric in its table entry for that destination to 1 (hop) more than the metric in that neighbor's update.

The primary improvement for ad hoc networks made in DSDV over standard distance vector routing is the addition of a *sequence number* in each routing table entry. The use of this sequence number prevents routing loops caused by updates being applied out of order; this problem may be common over multihop wireless transmission, since the routing information may spread along many different paths through the network.

Hash Chains: A one-way hash chain is built on a one-way hash function. Like a normal hash function, a one-way hash function, H, maps an input of any length to a fixed-length bit string. Thus, $H : \{0,1\}^* \rightarrow \{0,1\}^\rho$, where ρ is the length in bits of the output of the hash function. The function H should be simple to compute yet must be computationally infeasible in general to invert.

To create a one-way hash chain, a node chooses a random $x \in \{0,1\}^\rho$ and computes the list of values

$$h_0,\ h_1,\ h_2,\ h_3,\ \ldots,\ h_n$$

where $h_0 = x$, and $h_i = H(h_{i-1})$ for $0 < i \leq n$, for some n. The node at initialization generates the elements of its hash chain as shown above, from "left to right" (in order of increasing subscript i) and then over time uses certain elements of the chain to secure its routing updates; in using these values, the node progresses from "right to left" (in order of decreasing subscript i) within the generated chain.

Given an existing authenticated element of a one-way hash chain, it is possible to verify elements later in the sequence of use within the chain (further to the "left," or in order of decreasing subscript). For example, given an authenticated h_i value, a node can authenticate h_{i-3} by computing $H(H(H(h_{i-3})))$ and verifying that the resulting value equals h_i. To use one-way hash chains for authentication, we assume some mechanism for a node to distribute an authentic element such as h_n from its generated hash chain.

Authenticating Routing Updates: Each node in SEAD uses a specific single next element from its hash chain in each routing update that it sends about itself (metric 0). Based on this initial element, the one-way hash chain conceptually provides authentication for the lower bound of the metric in other routing updates for this destination; the authentication provides only a lower bound on the metric: An attacker can increase the metric, claim the same metric, but cannot decrease the metric.

We assume that an upper bound can be placed on the diameter of the ad hoc network, and we use $m-1$ to denote this bound. The method used by SEAD for authenticating an entry in a routing update uses the *sequence number* in that entry to determine a contiguous group of m elements from that destination node's hash chain, one element of which must be used to authenticate that routing update. The particular element from this group of elements that must be used to authenticate the entry is determined by the *metric* value being sent in that entry. Specifically, if a node's hash chain is the sequence of values

$$h_0,\ h_1,\ h_2,\ h_3,\ \ldots,\ h_n$$

and n is divisible by m, then for a sequence number i in some routing update entry, let $k = \frac{n}{m} - i$. An element from the group of elements

$$h_{km},\ h_{km+1},\ \ldots,\ h_{km+m-1}$$

from this hash chain is used to authenticate the entry; if the metric value for this entry is j, $0 \leq j < m$, then the value h_{km+j} here is used to authenticate the routing update entry for that sequence number. Nodes receiving any routing update can easily authenticate each entry in the update, given any earlier authentic hash element from the same hash chain.

Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks, *Yih-Chun Hu (Rice University), Adrian Perrig (UC Berkeley), David B. Johnson (Rice University)*

We describe some features of Ariadne [15], a secure on-demand routing protocol that withstands node compromise and relies only on highly efficient *symmetric* cryptography. Ariadne can authenticate routing messages using one of three schemes: shared secrets between each pair of nodes, shared secrets between communicating nodes combined with broadcast authentication, or digital signatures. We primarily discuss here the use of Ariadne with TESLA [30], an efficient broadcast authentication scheme that requires loose time synchronization. Using pairwise shared keys avoids the need for synchronization, but at the cost of higher key setup overhead.

Basic Ariadne Route Discovery: We present the design of the Ariadne protocol in two stages: we first present a mechanism that enables the target to verify the authenticity of the ROUTE REQUEST; we then present an efficient per-hop hashing technique to verify that no node is missing from the node list in the REQUEST. In the following discussion we assume that the initiator **S** performs a Route Discovery for target **D**, and that they share the secret keys K_{SD} and K_{DS}, respectively, for message authentication in each direction.

Target authenticates ROUTE REQUESTS*.* To convince the target of the legitimacy of each field in a ROUTE REQUEST, the initiator simply includes a MAC computed with key K_{SD} over unique data, for example a timestamp. The target can easily verify the authenticity and freshness of the route request using the shared key K_{SD}.

In a Route Discovery, the initiator wants to authenticate each individual node in the node list of the ROUTE REPLY. A secondary requirement is that the target can authenticate each node in the node list of the ROUTE REQUEST, so that it will return a ROUTE REPLY only along paths that contain only legitimate nodes. Each hop authenticates new information in the REQUEST. The target buffers the REPLY until intermediate nodes can release the corresponding TESLA keys. The TESLA security condition is verified at the target, and the target includes a MAC in the REPLY to certify that the security condition was met.

Per-hop hashing. Authentication of data in routing messages is not sufficient, as an attacker could remove a node from the node list in a REQUEST. We use one-way hash functions to verify that no hop was omitted, and we call this approach *per-hop hashing*. To change or remove a previous hop, an attacker must either hear a REQUEST without that node listed, or must be able to invert the one-way hash function.

Basic Ariadne Route Maintenance: Route Maintenance in Ariadne is based on DSR. A node forwarding a packet to the next hop along the source route returns a ROUTE ERROR to the original sender of the packet if it is unable to deliver the packet to the next hop after a limited number of retransmission attempts. In this section, we discuss mechanisms for securing ROUTE ERROR s, but we do not consider the case of attackers not sending ERROR s.

To prevent unauthorized nodes from sending ERROR s, we require that an ERROR be authenticated by the sender. Each node on the return path to the source forwards the ERROR. If the authentication is delayed, for example when TESLA is used, each node that will be able to authenticate the ERROR buffers it until it can be authenticated.

Avoiding Routing Misbehavior: The protocol described so far is vulnerable to an attacker that happens to be along the discovered route. In particular, we have not presented a means of determining whether intermediate nodes are in fact forwarding packets that they have been requested to forward. We choose routes based on their prior performance in packet delivery. Our scheme relies on feedback about which packets were successfully delivered. The feedback can be received either through an extra end-to-end network layer message, or by exploiting properties of transport layers, such as TCP with SACK [21]; this feedback approach is somewhat similar that used in IPv6 for Neighbor Unreachability Detection [24].

A node with multiple routes to a single destination can assign a fraction of packets that it originates to be sent along each route. When a substantially smaller fraction of packets sent along any particular route are successfully delivered, the node can begin sending a smaller fraction of its overall packets to that destination along that route.

Authenticated Routing for Ad Hoc Networks, *Kimaya Sanzgiri (UCSB), Bridget Dahill (UMass, Amherst), Brian N. Levine (UMass, Amherst), Clay Shields (Georgetown University, Washington DC), Elizabeth M. Belding-Royer (UCSB)*

Abstract: Most proposed routing protocols for mobile ad hoc networks are vulnerable to modification, impersonation and fabrication attacks. The proposed secure routing protocol, Authenticated Routing for Ad Hoc Networks, prevents such attacks through message authentication, integrity and non-repudiation. Simulation results show that ARAN maintains good network performance while offering significant security advantages over existing routing protocols.

Introduction: Mobile ad hoc networks consist purely of wireless mobile nodes with no wired infrastructure. Communication between nodes that are not within direct transmission range of each other is enabled through packet forwarding by intermediate nodes. The topology and membership of these networks is highly dynamic. Most of the proposed routing protocols for ad hoc networks are optimized for performance in such a dynamic environment. However, many of these proposed routing protocols have security vulnerabilities that may be exploited to launch different types of attacks.

In this analysis, three main categories of attacks are identified. The first of these are modification attacks, where malicious nodes can make illegitimate modifications to routing messages. The second category is that of impersonation or spoofing attacks, where a malicious node can fake its identity by illegally modifying its IP and/or MAC address in outgoing messages. Fabrication attacks form the third category of attacks, where a malicious node could inject false routing messages into the network. These techniques can be used both individually and in various combinations to cause illegal route redirection, route corruption and denial of service.

The proposed protocol, Authenticated Routing for Adhoc Networks (ARAN) [32], prevents the above types of attacks through message authentication, integrity and non-repudiation.

Protocol Description: The ARAN Protocol uses public key cryptography to guarantee message authentication, integrity and non-repudiation. The protocol is designed for the *managed-open* environment, where nodes can obtain a public key certificate from a common certification authority that is trusted by all other nodes in the environment. Typical examples of such an environment are classroom or conference scenarios. The operation of the protocol can be divided into route discovery and route maintenance phases.

The route discovery process is initiated by the source node by flooding a digitally signed Route Discovery packet (RDP) to its neighbors:

$$S \rightarrow \text{broadcast} : [\text{RDP}, \text{IP}_D, \text{cert}_S, N_S, t]K_{S-} \qquad (1)$$

When a neighbor A receives the RDP message, it sets up a reverse path back to the source node and verifies the signature of the source by extracting S's public key from its certificate. The node then signs the contents of the message, appends its own certificate, and broadcasts the message to its neighbors. When A's neighbor B receives the message, it validates A's signature, and then replaces it with its own signature (the signature of the source node is retained). The packet continues to be rebroadcast in this manner across the network until it reaches the destination.

When the first RDP reaches the destination, the destination node verifies the signature of the source node and then sends a digitally signed Route Reply packet (REP) back to the source. The REP travels along the same path as the RDP, and the same signing procedure is performed by intermediate nodes. Note that because the destination must sign the REP message, only the destination is allowed to respond to the RDP. Also, because RDP messages are signed at each hop and do not contain a hop count or a source route, malicious nodes have no opportunity to intentionally redirect traffic.

Route maintenance is performed through digitally signed Error messages that are initiated by the node directly upstream of a link failure.

Conclusion: ARAN provides both end-to-end and hop-by-hop authentication of route discovery and reply messages, preventing impersonation attacks. The digital signatures guarantee integrity and non-repudiation, preventing illegal message modification and enabling identification of the source of erroneous messages.

Simulation results show that ARAN provides the same throughput as leading ad hoc routing protocols with marginally increased overhead and delay due to the digital signatures.

Dynamic and Secure Group Membership in Ad Hoc and Peer-to-Peer Networks, *Claude Castelluccia (INRIA Rhône-Alpes) and Gabriel Montenegro (Sun Labs, Europe)*

Introduction: Ad hoc or peer-to-peer networks (called *impromptu* networks henceforth) pose many problems with respect to securing their highly dynamic structures. A naive approach assumes any given node can trust all other nodes in the impromptu network for any type of operation (for example, engaging in some cooperative and perhaps confidential activity). This paper improves on previous efforts to secure group authorization (including membership). We do so by employing crypto-based identifiers [23] for node and group identification, and then use these in authorization certificates. These allow groups (or nodes) to authorize nodes (or other groups) [10].

Our approach enables highly flexible and robust impromptu security services in an inherently distributed fashion. Previous work on securing impromptu networks has assumed the existence of a traditional PKI, of some web of trust or of some mechanism to distribute keys and shared secrets. We believe these assumptions are unrealistic in impromptu networks.

Secure Node Identity: First of all, we must start by defining an addressing model. Previous efforts for ad hoc networks conclude that since there is no aggregation to the degree possible with regular fixed networks, addressing can be more flexible. We propose to use pure identifiers with no topological meaning.

Our scheme improves upon these by having an implicit cryptographic binding between a node's identifier and its public key (or certificate). A node autoconfigures its (crypto-based) identifier (CBID) by doing the following:

- Create a pair of public and private keys (PK and SK).
- Create its CBID: $CBID = hash(PK)$.

Note that the hash function can be applied over more than just the public key (e.g., a salt or some other values) [23]. Given the secure correspondence between identity and public key, the latter can be communicated by the node itself. This simplifies key management, since no third parties need to be involved either in creating or distributing the public keys.

Provided the bit-length of the CBID's is large enough [23], these identifiers have two very important properties: (1) they are *statistically unique*, because of the collision-resistance property of the cryptographic hash function used to generate them, and (2) they are *securely bound to a given node*: the node can prove ownership of the CBID by signing packets with the corresponding private key. Any other node can verify the signature without relying on any centralized security service such as a PKI or Key Distribution Center.

These characteristics (1) make CBID's a very scalable naming system, well adapted to ad hoc environments, and (2) provide an autoconfigurable and solid foundation for nodes to engage in verifiable exchanges with each other.

Dynamic and Secure Node Authentication: The first application of the above is to protect basic exchanges between two peers or nodes in a network from malicious intermediate hosts. For example, in on-demand ad hoc routing protocols (e.g. [27, 26]), nodes discover each other by exchanging "route request" and "route reply" messages. We have recently shown how CBID can protect this basic exchange from impersonation attacks [9]. Similar work is underway for the JXTA[5] open-source peer-to-peer protocol.

Dynamic and Secure Group Membership: Subsequently, CBID's are used to express authorization via authorization certificates, similar to how they are used to *Secure Group Management for IPv6* [10].

Authorization certificates have the following form:

$$Cert = (group, node, delegation, tag, validity)$$

In the above, *group* is a group CBID for the entire impromptu network, or for a subset of it. Appropriately, the certificate is signed with the private key that corresponds to it. Here, *node* is the CBID of the beneficiary of this authorization, that is, the node that is authorized by the group to join it or perform certain services on its behalf. *delegation* is a boolean (in either SPKI [12] or KeyNote2 [2]) that specifies whether or not the group has allowed the *node* to further delegate the permission expressed in the next field. Finally, *tag* is the authorization to be a member of the signing group, or to perform certain services as authorized by the group.

This is an example of how a single node A with CBID of A_CBID could start an ad hoc network (really a group within a perhaps already physically existing ad hoc network) by following these steps:

- A creates the group group public and private key pair: G_PK and G_SK.
- A creates the group identifier: $G\_CBID = hash(G\_PK)$.
- A as the *group controller* issues a certificate to allow itself into the group: $(G\_CBID, A\_CBID, true, ``groupMembership'', someDuration)$.
- A as the *group controller* admits another node (e.g., B) into the group by issuing the corresponding certificate: $(G\_CBID, B\_CBID, false, ``groupMembership'', someDuration)$.

Now, either A or B can prove to other nodes that they are legitimate members of the group by sending a message which includes their certificate, and that is signed with their private key A_SK or B_SK, respectively.

Intrusion Detection, *Yongguang Zhang (HRL Labs & UT Austin) and Wenke Lee (GA Tech)*

Intrusion prevention measures, such as encryption and authentication, can be used in ad-hoc networks to reduce intrusions, but cannot eliminate them. For example, encryption and authentication cannot defend against compromised mobile nodes, which often carry the private keys. Insider attacks may also deem firewalls useless. The history of security research has taught us a valuable lesson – no matter how many intrusion prevention measures are inserted in a network, there are always some weak links that one could exploit to break in. In a high-survivability network, Intrusion Detection System (IDS) is necessary as a second wall of defense.

However, current IDS techniques are designed for wired networks only; they are inadequate for mobile ad-hoc networks (MANET). For example, today's network-based IDS relies on real-time traffic analysis of traces collected at switches, routers, or gateways. However, MANET environment does not have such traffic concentration point where one can collect audit data for the entire network. Instead, the only available audit trace is limited to communication activities taking place within the radio range. IDS is thus forced to work with this partial and localized information. Furthermore, thanks to mobility MANET is seemingly chaotic in nature, and this blends anomaly and normalcy situations together. There is nothing parallel to this in wired networks, and none of the current IDSes has been tested in such a dynamic environment.

Therefore, there are unique issues for IDS in MANET:

- A fully distributed solution. IDS must work alone at each node, but collaborative detection and investigation among neighboring nodes is possible.
- Localized and incomplete audit data. IDS algorithm may be required to sense anomaly hops away.
- Thin line between anomaly and normalcy, which requires an elaborated model to produce high detection rate with low false alarm rate.
- Resource constrains. IDS should not consume too much power as MANET environment is often operated on battery power.

Our goal is to develop IDS techniques that address these issues. In our preliminary study [35], we have developed a

[5] http://www.jxta.org

new architecture called “Distributed and Cooperative Intrusion Detection.” In this model, IDS agent runs at each mobile node and performs local data collection and local detection, whereas cooperative detection and global intrusion response can be triggered when a node reports anomaly. We have also proposed an anomaly detection model for detecting attacks on MANET routing protocols.

In a subsequent study [36], we have built such an IDS model and implemented it in a network simulator (ns2 with CMU’s MANET extension). We have also conducted a series of experiments to study its effectiveness. The experiments are on a 10-node MANET network. The performance evaluation is by measuring the detection rate and false-alarm rate (see Figure 2). We simulate three different types of intrusion conditions separately: route-attack model where a node’s route table is randomly falsified, traffic-attack model where a node randomly drops packets that it is supposed to forward, and “no-attack” (i.e., under normal use). In Figure 2, square, cylinder, and darker bars represent each of these conditions respectively. We also repeat the experiments with different MANET routing protocol (DSR, ADOV, or DSDV), and use two different types of IDS model-training tools (RIPPER or SVM-Light).

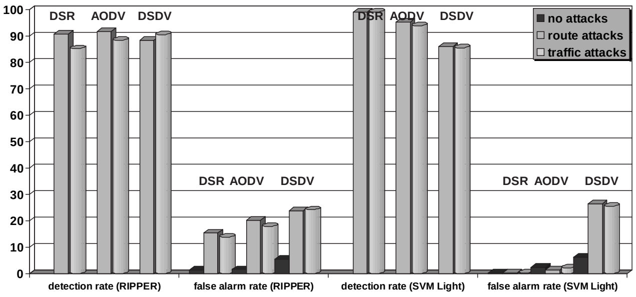

Figure 2: IDS performance measurement (Intrusion Detection, *Y. Zhang and W. Lee*)

The result seems rather positive: under some configuration (DSR + SVM-Light), our IDS model can detect 99% of the intrusions with lower than 1% false-alarm rate. More importantly, we do learn some useful lessons. First, there is indeed a very thin line between normalcy and anomaly in MANET. We had to try a large number of feature combinations and most of the earlier ones simply fell with low detection rate and high false alarm rate. The good news is that it is possible to find this line, like the final model we used to generate this result.

Another observation is the disparity among routing protocols. For example, on-demand protocols (DSR and AODV) are likely to out-perform table-driven ones (DSDV). This is due to the higher correlation in routing information, such as the connection between traffic pattern and route change in on-demand routing, and the source routes embedded in DSR. Therefore, when we design MANET protocol we should intentionally increase such correlation to help IDS.

Nonetheless, there are serious limitations to the IDS-in-MANET research in general. Intrusion detection heavily relies on the understanding of applications and threat models. Unfortunately, this research community does not yet know what constitute a “typical” application or scenario. We have used a mobility and application scenario very similar to what have been widely used in literature, such as 10 nodes moving in a random way-point pattern within a 1000×1000 space. This is artificial at best. IDS models developed using such scenario will hardly have any effectiveness in real life. So we believe the more pressing tasks are for us to better understand the potential applications for MANET, and to define realistic benchmarks. Without these, it will be very difficult to develop IDS techniques for use in MANET in the future.

Availability

Handling MAC Layer Misbehavior in Wireless Networks, *Pradeep Kyasanur (UIUC) and Nitin H. Vaidya (UIUC)*

Introduction: Wireless MAC protocols such as IEEE 802.11 use cooperative contention resolution mechanisms for sharing the channel. In this environment, some *selfish* hosts in the network can misbehave by failing to adhere to the network protocols with the intent of obtaining an unfair share of the channel bandwidth. Our work focuses on detecting and handling MAC layer misbehavior by *selfish* hosts in IEEE 802.11-based networks.

In IEEE 802.11 DCF mode, nodes exchange RTS and CTS packets to reserve the channel before data transmission (When data packets are small RTS/CTS exchange may be omitted.) A node with a packet to transmit picks a random backoff value b chosen uniformly from range $[0, CW]$, where CW is called the Contention Window, and transmits after waiting for b idle slots. If a transmission results in a collision, the CW value is doubled. The throughput obtained by a node is inversely proportional to the average time it waits in backing off. Therefore, misbehaving nodes can obtain a higher share of throughput by selecting small backoff values or by not doubling the CW value after a collision.

Handling Misbehavior: Misbehaving hosts may obtain an unfair share of channel bandwidth. Traffic analysis can be used to identify such misbehaving hosts. IEEE 802.11 protocol is unfair in the short-term and thus the monitoring interval should be sufficiently large for reasonably accurate misbehavior detection. Consequently, short-term misbehavior may not be detected using traffic analysis. In addition, a misbehaving node may restrict itself to a fair share of bandwidth and yet have low transmission delay by transmitting its packets with small backoff values. Traffic analysis may not detect such misbehavior. Game-theoretic techniques have also been used to develop MAC layer protocols that are resilient to misbehavior. However, these protocols may result in poorer throughput in the absence of misbehavior.

An alternate approach that we adopt is to modify the IEEE 802.11 protocol to simplify misbehavior detection while retaining its performance and fairness characteristics. We present a detection procedure [20] for a network having a well-behaved receiver and multiple potentially misbehaving senders. An example of this is an infrastructure-based network having a well-behaved base station (receiver) and multiple mobile hosts (senders) communicating with the base station. We use this example network to illustrate our solution. However, our solution may be applied to ad hoc networks as well.

Detection Procedure. The base station provides a backoff value b to be used by a host for its $(i+1)^{th}$ transmission to the base station in the CTS or ACK packet of the i^{th} transmission. In the event of a collision, the host generates a new backoff value using a deterministic function f with b as a parameter. The host includes the attempt number of retransmission in every RTS or DATA packet. When the base station receives a packet successfully, it computes the expected number of slots the host should have waited using the transmission attempt number and the originally assigned backoff value as parameters to the function f. The host is deemed to have deviated from the protocol if the number of idle slots sensed by the base station between the i^{th} and $(i+1)^{th}$ from the host is lesser than a fraction α of the expected number of slots. If K deviations are identified in a window of *THRESH* packets from the node (α, K and *THRESH* are protocol parameters), the host is designated to be *misbehaving*.

Correcting Misbehavior. The benefits gained by misbehaving nodes are increased throughput and decreased delay. The *correction* mechanism aims to negate these benefits without penalizing the conforming nodes. When the monitoring procedure detects that a host has waited for less than the expected backoff by an amount D, this amount D is added as penalty to the next backoff assigned to that node. In addition, a misbehaving node suffers fewer collisions per successful transmission on an average and we attempt to negate this benefit by adding additional penalty to the next backoff value assigned to that node.

Conclusion and Future Work: In this work, we have developed modifications to IEEE 802.11 protocol for simplified misbehavior detection in infrastructure-based networks. The approach may be used in ad hoc networks as well. Each node in the ad hoc network can monitor the traffic it receives to verify that the nodes sending the packets are well-behaved. We plan to augment our approach with mechanisms to detect a misbehaving node that gains more bandwidth by using multiple MAC addresses. We also plan to develop protocols to handle misbehavior for scenarios with misbehaving receivers, and scenarios with colluding senders and receivers. We will also explore other approaches for handling misbehavior.

Stimulating Cooperation of Selfish Nodes in MANET, *Pietro Michiardi (Inst. Eurecom) and Refik Molva (Inst. Eurecom)*

This paper focuses on a security issue specific to mobile ad hoc networks: node selfishness. Unlike networks using dedicated nodes to support basic functions like packet forwarding, routing, and network management, in MANET those functions are carried out by all available nodes. In such networks there is no good reason to assume that a node will cooperate and provide services to each other: service provision consumes energy, a scarce resource that nodes are induced to use for their own communications.

It is a realistic assumption that selfish nodes do not perform active attacks, due to the high energy consumption thereof. In the proposed security scheme (CORE), node cooperation is stimulated by a collaborative monitoring technique and a reputation mechanism. Each node of the network monitors the behavior of its neighbors with respect to a requested function and collects observations about the execution of that function: as an example, when a node initiates a Route Request (e.g., using the DSR routing protocol) it monitors that its neighbors process the request, whether with a Route Reply or by relaying the Route Request. If the observed result and the expected result coincide, then the observation will take a positive value, otherwise it will take a negative value.

Based on the collected observations, each node computes a reputation value for every neighbor. The formula used to evaluate the reputation value avoids false detections (caused for example by link breaks) by using an aging factor that gives more relevance to past observations: frequent variations on a node behavior are filtered. Furthermore, if the function that is being monitored provides an acknowledgement message (e.g., the Route Reply message of the DSR protocol), reputation information can also be gathered about nodes that are not within the radio range of the monitoring node. In this case, only positive ratings are assigned to the nodes that participated to the execution of the function in its totality.

The CORE mechanism resists to attacks performed using the security mechanism itself: no negative ratings are spread between the nodes, so that it is impossible for a node to maliciously decrease another node's reputation. The reputation mechanism allows the nodes of the MANET to gradually isolate selfish nodes: when the reputation assigned to a neighboring node decreases below a pre-defined threshold, service provision to the misbehaving node will be interrupted. Misbehaving nodes can, however, be reintegrated in the network if they increase their reputation by cooperating to the network operation.

An original approach to study the node selfishness problem in MANET is based on an economic model. In this model, service provision preferences for each node are represented by a utility function. As the name implies, the utility function quantifies the level of satisfaction a node gets from using the network resources. Game-theoretic methods are applied to study cooperation under this new model. Game theory is a powerful tool for modeling interactions between self-interested users and predicting their choice of strategy. Each player in the game maximizes some function of utility in a distributed fashion. The games settle at a Nash equilibrium if one exists. Since nodes act selfishly, the equilibrium point is not necessarily the best operating point from a social point of view. Pricing emerges as an effective tool for cooperation enforcement because of its ability to guide node behavior toward a more efficient operating point from the social point of view.

In our analysis we first identify the preference relations that are specific to our problem and then design a utility function that satisfies this structure. The utility function we used to model the selfishness problem takes into account the energy that a node spends for the purpose of its own communications and energy that the node has to use when participating in the routing protocol and when relaying data packets on behalf of other nodes. Node behavior is represented as the percentage of energy a node dedicates for its own communications and the percentage of energy spent for network operation (i.e., 50% means that the node uses half of its energy for itself, half for routing and packet forwarding). Simulations shows that under this definition, a selfish node that tries to maximize its utility function will dedicate the totality of its available energy for its own communications. The CORE mechanism is then modeled as the pricing function that is used to guide the operating point to a fair position: reputation influences the percentage of energy a node is allowed to use for its own communications. When the reputation rating is high, the amount of energy the node can spend for its own use rises and vice versa. Simulation results show a stable feedback system that converges to an operating point where 48% of the available energy is dedicated to the network operation (see Figure 3).

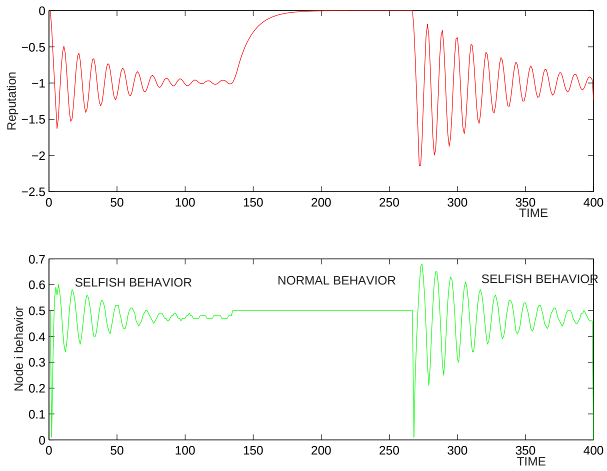

Figure 3: Reputation and node behavior in time (Stimulating Cooperation of Selfish Nodes in MANET, *P. Michiardi and R. Molva*)

Cooperation of Nodes, *Sonja Buchegger (IBM Research, Zürich) and Jean-Yves Le Boudec (EPFL)*

In game-theoretic terms, cooperation is a dilemma. The dominating strategy for individual nodes is not to cooperate, as cooperation consumes resources and it might result in a disadvantage. But if every node follows that strategy, the outcome is undesirable for everyone as it results in a non functional or entirely absent network.

Learning by Observing – CONFIDANT: Our approach is to find the selfish and/or malicious nodes and to isolate them, so that misbehavior will not pay off but result in isolation and thus cannot continue. CONFIDANT [4] is short for ‘Cooperation Of Nodes, Fairness In Dynamic Ad-hoc NeTworks’ and detects malicious nodes by means of observation or reports about several types of attacks, thus allowing nodes to route around misbehaved nodes and to isolate them. Figure 4 shows the CONFIDANT components as extension to a routing protocol such as Dynamic Source Routing (DSR).

Nodes have a *monitor* for observations, *reputation records* for first-hand and trusted second-hand observations about routing and forwarding behavior of other nodes, *trust records* to control trust given to received warnings, and a *path manager* to adapt their behavior according to reputation and to take action against malicious nodes. The term *reputation* is used to evaluate routing and forwarding behavior according to the network protocol, whereas the term *trust* is used to evaluate participation in the CONFIDANT meta-protocol.

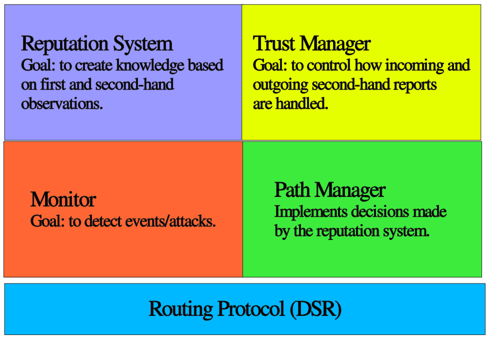
Reputation System
Goal: to create knowledge based on first and second-hand observations.
Trust Manager
Goal: to control how incoming and outgoing second-hand reports are handled.
Monitor
Goal: to detect events/attacks.
Path Manager
Implements decisions made by the reputation system.
Routing Protocol (DSR)

Figure 4: CONFIDANT Components in a Node (Cooperation of nodes, *S. Buchegger and J.-Y. Le Boudec*)

The dynamic behavior of CONFIDANT is as follows. Nodes *monitor* their neighbors and change the *reputation* accordingly. If they have reason to believe that a node misbehaves, they can take *action* in terms of their own routing and forwarding and they can decide to inform other nodes by sending an ALARM message. When a node receives such an ALARM either directly or by promiscuously listening to the network, it evaluates how trustworthy *trust* the ALARM is based on the source of the ALARM and the accumulated ALARM messages about the node in question. It can then decide whether to take *action* against the misbehaving node.

Simulations for “no forwarding” have shown that CONFIDANT can cope well, even if half of the network population acts maliciously.

Robustness: Assuming that nodes employ the CONFIDANT protocol, malicious nodes can be detected and isolated. However, we have to ensure the robustness of CONFIDANT itself, for example its robustness against wrong observations, i.e., categorizing behavior as an attack when it is not or vice versa, and its robustness against wrong accusations, i.e., maliciously excluding cooperative nodes by spreading the rumor that they misbehave. To facilitate categorization and trust management, we use a Bayesian approach, in which the belief of a node about its environment as captured in the reputation records is updated at each observation. The belief models of several nodes are then compared, evaluating the compatibility and excluding outliers.

This assumes that nodes that spreading wrong accusations are not in the majority at any given time.

Conclusions: Our goals are to increase cooperation by proactively giving selfish nodes an incentive to cooperate, as well as reactively isolate selfish or malicious nodes such that they cannot continue their misbehavior. To make cooperation in mobile ad-hoc networks attractive we have to make sure that selfish behavior, i.e., a behavior that maximizes the utility of a node, leads to an outcome that is also beneficial for the network. In CONFIDANT this is achieved by gradually isolating a node that has accumulated a bad reputation, such that, —once detected— it can no longer benefit from the network, so if a node wants to remain in the network it has to cooperate. This effect is also exploited to largely neutralize malicious nodes when other nodes no longer route or forward with or for them, by reducing their influence to the radius of one hop. We are currently investigating how to use the trust administration to provide incentives to cooperate also at the meta-level, i.e., to participate in CONFIDANT.

Stimulating Cooperation by Means of *Nuglets*, *Levente Buttyán (EPFL) and Jean-Pierre Hubaux (EPFL)*

In civilian applications of ad hoc networks, where each node is its own authority, nodes may selfishly deny cooperation in order to save their own resources (e.g., battery power, memory, CPU cycles).

One approach to solving this problem would be to make the nodes tamper resistant, so that their behavior cannot be modified by their users. However, this approach does not seem to be very realistic, since ensuring that the whole device is tamper resistant may be very difficult, if not impossible. Therefore, we propose another approach that requires only a tamper resistant hardware module (such as the SIM card in GSM phones), called *security module*, in each node. Under the assumption that the user can possibly modify the behavior of the node, but never that of the security module, our design ensures that tampering with the node is *not advantageous* for the user, and therefore, it should happen only rarely.

We focus on the stimulation of packet forwarding, which is a fundamental networking function that the nodes should perform. In a nutshell, we propose a protocol that requires the node to pass each packet (generated as well as received for forwarding) to its security module. The security module maintains a counter, called *nuglet counter*. When the node wants to send a packet as originator, the number n of forwarding nodes that are needed to reach the destination is estimated, and the nuglet counter is decreased by n. When the node forwards a packet, its nuglet counter is increased by one. The value of the nuglet counter must remain positive, which means that if the node wants to send its own packets, then it must forward packets for the benefit of other nodes. The nuglet counter is protected from illegitimate manipulation by the tamper resistance of the security module.

In order to study the behavior of the proposed protocol, we conducted simulations written in plain C++ language. In our simulations, each node generates packets with a constant average rate. If an own packet cannot be sent (due to the low value of the nuglet counter), then it is dropped. We model selfishness of the nodes by the goal of minimizing the number of own packets dropped, which is equivalent to maximizing $z_o = \frac{out_o}{in_o}$, where out_o denotes the number of own packets sent by the node and in_o stands for the number of own packets generated by the node during the simulation time.

We studied the performance of the following three forwarding rules:

- *Rule 1*: always forward
- *Rule 2*: if $c \leq C$ then forward else forward with probability C/c and drop with probability $1 - C/c$
- *Rule 3*: if $c \leq C$ then forward else drop

where c and C denote the current and the initial number of nuglets at the node, respectively. Clearly, the most cooperative rule is Rule 1; Rules 2 and 3 are less cooperative, in this order.

We set 90% of the nodes to use a given rule (we call this the *majority rule*), and the remaining 10% of the nodes to use Rule 1 in a first set of simulations, Rule 2 in a second set of simulations, and finally Rule 3 in a third set of simulations. We observed the average value of z_o that the 10% deviating nodes could achieve in each case. The results are shown in Table 2.

Majority rule	Average value of z_o of the 10% deviating nodes when they use:		
	Rule 1	Rule 2	Rule 3
Rule 1	0.979	0.935	0.924
Rule 2	0.941	0.905	0.897
Rule 3	0.898	0.873	0.865

Table 2: Comparison of forwarding rules (Stimulating cooperation by means of *nuglets*, *L. Buttyan and J.-P. Hubaux*)

Remarkably, Rule 1 performed the best in every case. This means that nodes drop the smallest portion of their own packets when they use Rule 1, no matter whether the majority of the nodes use Rule 1, Rule 2, or Rule 3. Furthermore, this is true for every packet generation rate that we have simulated. Therefore, we conclude that the proposed mechanism indeed stimulates the nodes for packet forwarding.

A full description of the proposed approach, its protection scheme, and the simulations can be found in [6].

Cryptographic protocols

Cryptography with Guardian Angels: Bringing Civilization to Pirates, *Gildas Avoine (EPFL) and Serge Vaudenay (EPFL)*

Abstract: In contrast with traditional cryptographic protocols in which parties can have access to common third parties, and where at least one of them is assumed to be honest, we propose here a new model which is relevant for networks of communication devices with security modules. We then focus on the problem of fair exchange in this model. We propose a probabilistic protocol which provides arbitrarily low unfairness (involving a complexity cost).

Pirates and Guardian Angels Model: In classical mobile networks, the communication scheme is made of both devices P_i and providers; the latter may have put security modules G_i in their devices. G_i's can only communicate with their own P_i, and P_i's can only communicate with both their security module and their provider.

In future networks like self-organized mobile networks, the communication chain looks as follows and providers are no longer involved.

$$\begin{tikzcd}
P_1 \arrow[r, leftrightarrow] \arrow[d, leftrightarrow] & P_2 \arrow[r, leftrightarrow] \arrow[d, leftrightarrow] & \ldots \arrow[r, leftrightarrow] & P_{n-1} \arrow[r, leftrightarrow] \arrow[d, leftrightarrow] & P_n \arrow[d, leftrightarrow] \\
G_1 & G_2 & & G_{n-1} & G_n
\end{tikzcd}$$

Here we may assume that all users try to optimize their benefit and are potential pirates, and thus security modules serve as Guardian Angels in order to enforce community rules. Since users can modify the behavior of their devices, we can consider a user and a device as a same entity, a Pirate.

For practical considerations, we assume that guardians are tamper-resistant, simple and limited devices, and that they can only communicate with their own device P. Smart cards are examples of guardian angels. Guardians are set up by a given provider who may define his own community bylaw. This setting may lead to some interesting business models.

In this model, confidentiality, integrity, and authenticity are addressed in a classical way. A specific problem is the insurance of receipt: how to certify that a message was well transmitted and received? Here we address this problem in context of fair exchange problem.

Fair Exchange Protocol: The proposed protocol is a probabilistic fair exchange protocol between two entities P_i and P_{i+1} without trusted third party, which is in this way particularly relevant for self-organized networks. We recall first that an exchange protocol is fair if, at the end of the execution, either both parties have received the expected value v_i and v_{i+1}, or none of them have received any information about the other value.

In the pirates and angels model, we assume that G_i's have a virtual private network (VPN) which protects confidentiality, integrity, authentication, and sequentiality of the exchanged messages. So, the only possible attack consists in aborting the protocol. During a first stage, G_i and G_{i+1} use this VPN in order to exchange the expected values v_i and v_{i+1}.

The remaining stage is a simple synchronization problem: G_i and G_{i+1} need to decide in a synchronous way that the exchange succeeded in order to disclose the exchanged value to their devices. In this synchronization protocol, the guardians, in turn, send a message through the VPN. This message can be a termination signal or some dummy random value. To do this, they flip a coin with probability p to issue a termination signal. The pirates P_i and P_{i+1} are assumed to be unable to distinguish the signal from a random value. Then, guardians consider that the protocol succeeds when they have both received and sent the termination signal. Since pirates are assumed to get no information in the messages, they cannot decide to stop depending on the protocol view. Hence they must decide to stop at some level no matter what the communication are.

An analysis of our synchronization protocol yields to the following theorems:

Theorem 1: Let C be the number of messages exchanged between G_i and G_{i+1}. If p is the termination probability, we have $E(C) = \frac{3}{p} - 1$ when P_i and P_{i+1} are honest.

Theorem 2 : If p is the termination probability and p_a the highest probability of unfairness over all possible misbehavior of pirates, we have $\frac{p}{4} \leq p_a \leq \frac{p}{4(1-p)}$.

The proofs of these theorems are available in the extended abstract, as well as variants and extensions.

Rational Exchange, *Levente Buttyán (EPFL) and Jean-Pierre Hubaux (EPFL)*

There are many applications where two parties have to exchange digital items via a network. Clearly, each party would like to have some guarantee that the other party will not bring her in a disadvantageous situation by misbehaving in the exchange. The usual solution to this problem is to use a fair exchange protocol, which ensures for a correctly behaving party that it cannot suffer any disadvantages. However, fair exchange is impossible without a trusted third party, and therefore, its implementation can be problematic in infrastructureless ad hoc networks.

A promising approach to solve the exchange problem in ad hoc networks is based on the concept of *rational exchange*. A rational exchange protocol ensures that a misbehaving party cannot gain anything with the misbehavior. Therefore, rational (self-interested) parties have no reason to misbehave. However, unlike in fair exchange, a correctly behaving party may suffer a disadvantage. While rational exchange seems to provide weaker guarantees than fair exchange does, it has an appealing feature: under certain assumptions, rational exchange is possible without a trusted third party.

In order to be able to design rational exchange protocols, one needs to have a fairly good understanding of the concept. However, currently, rational exchange is not well understood, and often confused with fair exchange. Our goal is, therefore, to clarify this situation. In order to do so, we propose a formal model of exchange protocols, which is based on game theory [5]. We model the situation in which parties of a given exchange protocol find themselves as a game. We call this game the protocol game. The protocol game encodes all the possible interactions of the protocol parties. The protocol parties are modeled as players. The protocol itself (as a set of rules) is represented as a set of strategies (one strategy for each protocol party). Misbehavior means that a protocol party follows a strategy that is different from its prescribed strategy.

We define the concept of rational exchange in terms of properties of the protocol game and the prescribed strategies of the protocol parties. More precisely, we have been inspired by the striking similarity between the informal definition of rational exchange and the concept of Nash equilibrium in games. Therefore, we define rational exchange formally in terms of a Nash equilibrium in the protocol game:

Definition: Let us consider a two-party exchange protocol $\pi = \{\pi_1, \pi_2\}$, where π_1 and π_2 are the programs for the protocol parties. Furthermore, let us consider the protocol game G_π of π, and let us denote the strategy of player p_k that represents π_k within G_π by $s^*_{p_k}$, $k \in \{1, 2\}$. π is said to be rational *iff*

- *$(s^*_{p_1}, s^*_{p_2})$ is a Nash equilibrium in the protocol game G_π; and*
- *both p_1 and p_2 prefer the outcome of $(s^*_{p_1}, s^*_{p_2})$ to the outcome of any other Nash equilibrium in G_π.*

Our model is sufficiently rich to permit the definition of other properties of exchange protocols as well. More specifically, we can also define fairness. Representing the concepts of rational exchange and fair exchange in the same model allows us to study their relationships. In particular, we prove that fairness implies rationality (assuming that the protocol satisfies certain additional requirements):

Theorem: If the protocol satisfies the effectiveness, gain closed, and safe back out properties, then fairness implies rationality.

It is easy to see that the reverse is not true in general. Thus, the result that we obtain from the model justifies the intuition that fairness is a stronger requirement than rationality.

Besides leading to a deeper understanding of the concept of rational exchange and its relationship to fair exchange, our formalism can also be used to formally verify existing rational exchange protocols. We illustrate this by analyzing a rational exchange protocol proposed in the literature by Syverson [7]. Our analysis leads to the conclusion that Syverson's protocol is rational only under the assumption that the network is reliable, which means that it delivers messages within a constant time interval; if this assumption is removed, then the rationality property is lost.

This work is relevant in the context of ad hoc networking as rational exchange might be the only viable solution to the exchange problem if the network is fully self-organizing. To the best of our knowledge, we are the first who formalized the concept of rational exchange in its full generality, studied its relation to fair exchange, and provided rigorous proofs of rationality for existing rational exchange protocols. Perhaps, the most original contribution of this work is the usage of game theory as a tool for modeling and analyzing security protocols. It shows that game theory can successfully be used for such a task. In fact, we believe that it is the most appropriate tool for modeling rational exchange in particular.

References

[1] B. Bellur and R. G. Ogier. A reliable, efficient topology broadcast protocol for dynamic networks. In *Proceedings of the Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies (Infocom)*, pages 178–186, March 1999.

[2] M. Blaze, J. Feigenbaum, J. Ioannidis, and A. Keromytis. The KeyNote trust-management system Version 2. Internet RFC 2704, September 1999.

[3] E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm Intelligence: From Natural to Artificial Systems*. SFI Studies in the Sciences of Complexity. Oxford University Press, July 1999.

[4] S. Buchegger and J.-Y. Le Boudec. Performance analysis of the CONFIDANT protocol. In *Proceedings of the 3^{rd} ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, pages 226–236, June 2002.

[5] L. Buttyán and J.-P. Hubaux. Rational exchange – a formal model based on game theory. In *Proceedings of the 2^{nd} International Workshop on Electronic Commerce (WELCOM)*, November 2001.

[6] L. Buttyán and J.-P. Hubaux. Stimulating cooperation in self-organizing mobile ad hoc networks. *ACM/Kluwer Mobile Networks and Applications (MONET)*, to appear 2002.

[7] L. Buttyán, J.-P. Hubaux, and S. Čapkun. A formal analysis of Syverson's rational exchange protocol. In *Proceedings of the 15^{th} IEEE Computer Security Foundations Workshop (CSFW)*, June 2002.

[8] S. Čapkun, L. Buttyán, and J.-P. Hubaux. Small worlds in security systems: an analysis of the PGP certificate graph. In *Proceedings of the ACM New Security Paradigms Workshop*, 2002.

[9] C. Castelluccia and G. Montenegro. Protecting AODVng against impersonation attacks. *ACM Mobile Computing and Communications Review*, July 2002.

[10] C. Castelluccia and G. Montenegro. Securing group management in IPv6. Technical report, INRIA, August 2002.

[11] M. Corner and B. Noble. Zero-interaction authentication. In *Proceedings of the 8^{th} ACM International Conference on Mobile Computing and Networking (MobiCom)*, September 2002.

[12] C. Ellison and al. SPKI certificate theory. Internet RFC 2693, September 1999.

[13] Y.-C. Hu, D. B. Johnson, and A. Perrig. Secure efficient distance vector routing in mobile wireless ad hoc networks. In *Proceedings of the 4^{th} IEEE Workshop on Mobile Computing Systems and Applications* (WMCSA), June 2002.

[14] Y.-C. Hu, A. Perrig, and D. B. Johnson. Packet leashes: A defense against wormhole attacks in wireless ad hoc networks. Technical Report TR01-384, Department of Computer Science, Rice University, December 2001.

[15] Y.-C. Hu, A. Perrig, and D. B. Johnson. Ariadne: A secure on-demand routing protocol for ad hoc networks. In *Proceedings of the 8^{th} ACM International Conference on Mobile Computing and Networking (MobiCom)*, September 2002.

[16] J.-P. Hubaux, L. Buttyán, and S. Čapkun. The quest for security in mobile ad hoc networks. In *Proceedings of the 2^{nd} ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, October 2001.

[17] D. B. Johnson. Routing in ad hoc networks of mobile hosts. In *Proceedings of the IEEE Workshop on Mobile Computing Systems and Applications* (WMCSA), pages 158–163, December 1994.

[18] T. Kindberg, K. Zhang, and N. Shankar. Context authentication using constrained channels. In *Proceedings of the IEEE Workshop on Mobile Computing Systems and Applications* (WMCSA), pages 14–21, June 2002.

[19] J. Kong, P. Zerfos, H. Luo, S. Lu, and L. Zhang. Providing robust and ubiquitous security support for manet. In *Proceedings of the 9^{th} IEEE International Conference on Network Protocols (ICNP)*, 2001.

[20] P. Kyasanur and N. H. Vaidya. Detection and handling of mac layer misbehavior in wireless networks. Technical report, CSL, UIUC, August 2002.

[21] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow. TCP selective acknowledgment options. Internet RFC 2018, October 1996.

[22] R. Merkle. Protocols for public key cryptosystems. In *Proceedings of the IEEE Symposium on Security and Privacy*, 1980.

[23] G. Montenegro and C. Castelluccia. Statistically Unique and Cryptographically Verifiable (SUCV) identifiers and addresses. In *Proceedings of the 9th Annual Network and Distributed System Security Symposium (NDSS)*, February 2002.

[24] T. Narten, E. Nordmark, and W. A. Simpson. Neighbor discovery for IP version 6 (IPv6). Internet RFC 2461, December 1998.

[25] P. Papadimitratos and Z. J. Haas. Secure routing for mobile ad hoc networks. In *Proceedings of SCS Communication Networks and Distributed Systems Modeling and Simulation (CNDS)*, January 2002.

[26] C. Perkins, E. Belding-Royer, and S. Das. Ad hoc on demand distance vector (AODV) routing for IP version 6. Internet Draft, draft-perkins-manet-aodv6-01.txt, November 2001.

[27] C. Perkins, E. Belding-Royer, and S. Das. Ad hoc on demand distance vector (AODV) routing, June 2002.

[28] C. Perkins and P. Bhagwat. Highly dynamic Destination-Sequenced Distance-Vector routing (DSDV) for mobile computers. In *Proceedings of the ACM SIGCOMM Conference on Communication Architectures, Protocols, and Applications*, pages 234–244, August 1994.

[29] C. Perkins and E. Royer. Ad-hoc on-demand distance vector routing. In *Proceedings of the IEEE Workshop on Mobile Computing Systems and Applications* (WMCSA), pages 90–100, February 1999.

[30] A. Perrig, R. Canetti, D. Tygar, and D. Song. Efficient authentication and signature of multicast streams over lossy channels. In *Proceedings of the IEEE Symposium on Security and Privacy*, pages 56–73, May 2000.

[31] A. Qayyum, L. Viennot, and A. Laouiti. Multipoint relaying: An efficient technique for flooding in mobile wireless networks. Technical Report RR-3898, INRIA, February 2000.

[32] K. Sanzgiri, B. Dahill, B. N. Levine, C. Shields, and E. M. Belding-Royer. A secure routing protocol for ad hoc networks. In *Proceedings of the 10^{th} IEEE International Conference on Network Protocols (ICNP)*, November 2002.

[33] B. Schneier. *Secrets and Lies: Digital Security in a Networked World*. Wiley Computer Publishing, 2000.

[34] Trimble Navigation Limited. Data sheet and specifications for Trimble Thunderbolt GPS Disciplined Clock. Sunnyvale, California. Available at `http://www.trimble.com/thunderbolt.html`.

[35] Y. Zhang and W. Lee. Intrusion detection in wireless ad-hoc networks. In *Proceedings of the 6^{th} ACM International Conference on Mobile Computing and Networking (MobiCom)*, August 2000.

[36] Y. Zhang, W. Lee, and Y. Huang. Intrusion detection techniques for mobile wireless networks. *ACM/Kluwer Mobile Networks and Applications (MONET)*, to appear 2002.

[37] L. Zhou and Z. Haas. Securing ad hoc networks. *IEEE Network*, 13(6):24–30, Nov/Dec 1999.

[38] L. Zhou, F. B. Schneider, and R. van Renesse. COCA: A secure distributed on-line certification authority. *ACM Transactions on Computer Systems*, to appear.