Security in Ad hoc Networks

Refik Molva and Pietro Michiardi

1. Introduction

A mobile *ad hoc* network (MANET) consists of a set of mobile hosts that carry out basic networking functions like packet forwarding, routing, and service discovery without the help of an established infrastructure. Nodes of an ad hoc network rely on one another in forwarding a packet to its destination, due to the limited range of each mobile host's wireless transmissions. Security in MANET is an essential component for basic network functions like packet forwarding and routing: network operation can be easily jeopardized if countermeasures are not embedded into basic network functions at the early stages of their design. Unlike networks using dedicated nodes to support basic functions like packet forwarding, routing, and network management, in ad hoc networks those functions are carried out by all available nodes. This very difference is at the core of the security problems that are specific to ad hoc networks. As opposed to dedicated nodes of a classical network, the nodes of an ad hoc network cannot be trusted for the correct execution of critical network functions. These security problems call on the other hand for different solutions based on the organizational links between the nodes of a MANET:

- in managed environments, the nodes are controlled by an organization (or a structured set of organizations) and an *a priori trust* relationship between the nodes can be derived from the existing trust relationship of the organization;
- in open environments whereby nodes and their owners aren't linked by any organizational relationship, network security mechanisms cannot rely on any existing trust relationship among the nodes.

In managed environments, entity authentication can be sufficient to verify the trust level of each node in the organization and correct execution of critical network functions is assured based on the organizational trust. Such a priori trust can only exist in a few special scenarios like military networks and corporate networks, where a common, trusted authority manage the network, and requires tamper-proof hardware for the implementation of critical functions. Entity authentication in a large network on the other hand raises key management requirements.

In managed environments without tamper-proof hardware and strong authentication infrastructure, or in open environments where a common authority that regulates the network does not exist, any node of an ad hoc network can endanger the reliability of basic functions like routing. The correct operation of the network requires not only the correct execution of critical network functions by each participating node but it also requires that each node performs a fair share of the functions. The latter requirement seems to be a strong limitation for wireless mobile nodes whereby power saving is a major concern. The threats considered in the MANET scenario are thus not limited to maliciousness and a new type of misbehavior

called selfishness should also be taken into account to prevent nodes that simply do not cooperate.

With lack of a priori trust, classical network security mechanisms based on authentication and access control cannot cope with selfishness and cooperative security schemes seem to offer the only reasonable solution. In a cooperative security scheme, node misbehavior can be detected through the collaboration between a number of nodes assuming that a majority of nodes do not misbehave.

The remainder of this article presents security threats and major solutions from the literature along routing, cooperation enforcement and key management in MANET.

2. Routing Security in MANET

Unlike traditional networks whereby routing functions are performed by dedicated nodes or routers, in MANET, routing functions are carried out by all available nodes. Likewise, common routing security mechanisms consist of node and message authentication referring to an priori trust model in which legitimate routers are believed to perform correct operations. Authentication of a node or its messages does not guarantee the correct execution of routing functions in open environments with lack of a priori trust like MANET.

Security exposures of ad hoc routing protocols are due to two different types of attacks: *active* attacks through which the misbehaving node has to bear some energy costs in order to perform some harmful operation and *passive attacks* that mainly consist of lack of cooperation with the purpose of energy saving. Nodes that perform active attacks with the aim of damaging other nodes by causing network outage are considered to be *malicious* while nodes that perform passive attacks with the aim of saving battery life for their own communications are considered to be *selfish*.

Malicious nodes can disrupt the correct functioning of a routing protocol by *modifying* routing information, by *fabricating* false routing information and by *impersonating* other nodes. Recent research studies [10] brought up also a new type of attack that goes under the name of *wormhole* attack. On the other side, selfish nodes can severely degrade network performances and eventually partition the network [11] by simply not participating to the network operation.

In the existing ad hoc routing protocols nodes are trusted in that they do not maliciously tamper with the content of protocol messages transferred among nodes. Malicious nodes can easily perpetrate *integrity attacks* by simply altering protocol fields in order to subvert traffic, deny communication to legitimate nodes (denial of service) and compromise the *integrity* of routing computations in general. As a result the attacker can cause network traffic to be dropped, redirected to a different destination or to take a longer route to the destination increasing communication delays. A special case of integrity attacks is *spoofing* whereby a malicious node impersonates a legitimate node due to the lack of authentication in the current ad hoc routing protocols. The main result of spoofing attacks is the misrepresentation of the network topology that possibly causes network loops or partitioning. Lack of integrity and authentication in routing protocols can further be exploited through “fabrication” referring to the generation of bogus routing messages. Fabrication attacks cannot be

detected without strong authentication means and can cause severe problems ranging from denial of service to route subversion.

A more subtle type of active attack is the creation of a tunnel (or wormhole) in the network between two colluding malicious nodes linked through a private connection by-passing the network. This exploit allows a node to short-circuit the normal flow of routing messages creating a virtual vertex cut in the network that is controlled by the two colluding attackers.

Another exposure of current ad hoc routing protocols is due node selfishness that result in lack of cooperation among ad hoc nodes. A selfish node that wants to save battery life for its own communication can endanger the correct network operation by simply not participating in the routing protocol or by not forwarding packets as in the so called black hole attack. Current ad hoc routing protocols do not address the selfishness problem.

3. Secure Routing Proposals

Current efforts towards the design of secure routing protocols are mainly oriented to reactive (on-demand) routing protocols such as DSR [12] or AODV [13], where a node attempts to discover a route to some destination only when it has a packet to send to that destination. On-demand routing protocols have been demonstrated to perform better with significantly lower overheads than proactive routing protocols in many scenarios since they are able to react quickly to topology changes while keeping routing overhead low in periods or areas of the network in which changes are less frequent. It is possible to find, however, interesting security solutions for proactive routing protocols which are worthwhile to mention.

Current secure routing protocols proposed in the literature take into account *active attacks* performed by malicious nodes that aim at intentionally tampering with the execution of routing protocols whereas *passive attacks* and the selfishness problem are not addressed. Furthermore the prerequisite for all the available solutions is a *managed* environment characterized by some security infrastructure established prior to the secure routing protocol execution. The most significant proposals for secure routing in ad hoc networks are outlined in the sequel of this section.

3.1. Secure Routing Protocol

The Secure Routing Protocol (SRP) [1] is designed as an extension compatible with a variety of existing *reactive* routing protocols. SRP combats attacks that disrupt the route discovery process and guarantees the acquisition of correct topological information: SRP allows the initiator of a route discovery to detect and discard bogus replies. SRP relies on the availability of a *security association* (SA) between the source node (S) and the destination node (T). The SA could be established using a hybrid key distribution based on the public keys of the communicating parties. S and T can exchange a secret symmetric key ($K_{S,T}$) using the public keys of one another to establish a secure channel. S and T can then further proceed to mutual authentication of one another and the authentication of routing messages.

SRP copes with non-colluding *malicious* nodes that are able to modify (corrupt), replay and fabricate routing packets. In case of the Dynamic Source Routing (DSR) protocol [12], SRP requires including a 6-word header containing unique identifiers that tag the discovery process and a message authentication code (MAC) computed using a keyed hash algorithm. In order to initiate a route request (RREQ) the source node has to generate the MAC of the entire IP header, the basic protocol RREQ packet and the shared key $K_{S,T}$.

The intermediate nodes that relay the RREQ towards the destination measure the frequencies of queries received from their neighbors in order to regulate the query propagation process: each node maintains a priority ranking that is inversely proportional to the query rate. A node that maliciously pollutes network traffic with unsolicited RREQ will be served last (or ignored) because of its low priority ranking.

Upon reception of a RREQ, the destination node verifies the *integrity* and *authenticity* of the RREQ by calculating the keyed hash of the request fields and comparing them with the MAC contained in the SRP header. If the RREQ is valid, the destination initiates a route replay (RREP) using the SRP header the same way the source did when initiating the request. The source node discards replays that do not match with pending query identifiers and checks the integrity using the MAC generated by the destination.

The basic version of SRP suffers from the route cache poisoning attack: routing information gathered by nodes that operate in promiscuous mode in order to improve the efficiency of the DSR protocol could be invalid, because of potential fabrication by malicious nodes. The authors propose two alternative designs of SRP that use an Intermediate Node Reply Token (INRT). INRT allows intermediate nodes that belong to the same group that share a common key (K_G) to validate RREQ and provide valid RREP messages.

SRP suffers also from the lack of a validation mechanism for route maintenance messages: route error packets are not verified. However, in order to minimize the effects of fabricated error messages, SRP source-routes error packets along the prefix of the route reported as broken: the source node can thus verify that each route error feedback refers to the actual route and that it was originated at the a node that is part of the route. A malicious node can harm only routes it actually belongs to.

Assuming that the neighbor discovery mechanism maintains information on the binding of the medium access control and the IP addresses of nodes, SRP is proven to be essentially immune to IP spoofing [1].

SRP is, however, not immune to the wormhole attack: two colluding malicious nodes can misroute the routing packets on a private network connection and alter the perception of the network topology by legitimate nodes.

3.2. ARIADNE

Hu, Perrig and Johnson present an *on-demand* secure ad hoc routing protocol based on DSR that withstands node compromise and relies only on highly efficient *symmetric* cryptography. ARIADNE guarantees that the target node of a route discovery process can authenticate the initiator, that the initiator can authenticate each intermediate node on the path to the destination present in the RREP message and that

no intermediate node can remove a previous node in the node list in the RREQ or RREP messages.

As for the SRP protocol, ARIADNE needs some mechanism to bootstrap authentic keys required by the protocol. In particular, each node needs a shared secret key ($K_{S,D}$, is the shared key between a source S and a destination D) with each node it communicates with at a higher layer, an authentic TESLA [3, 4] key for each node in the network and an authentic "Route Discovery chain" element for each node for which this node will forward RREQ messages.

ARIADNE provides point-to-point *authentication* of a routing message using a message authentication code (MAC) and a shared key between the two parties. However, for authentication of a broadcast packet such as RREQ, ARIADNE uses the TESLA broadcast authentication protocol. ARIADNE copes with attacks performed by *malicious* nodes that modify and fabricate routing information, with attacks using impersonation and, in an advanced version, with the wormhole attack. Selfish nodes are not taken into account.

In ARIADNE, the basic RREQ mechanism is enhanced by eight additional fields used to provide authentication and integrity to the routing protocol as follows:

<ROUTE REQUEST, initiator, target, id, time interval, hash chain, node list, MAC list>

The initiator and target are set to the address of the initiator and target nodes, respectively. As in DSR, the initiator sets the id to an identifier that it has not recently used in initiating a Route Discovery. The time interval is the TESLA time interval at the pessimistic expected arrival time of the request at the target, accounting for clock skew. The initiator of the request then initializes the hash chain to $MAC_{KS,D}$ (initiator, target, id, time interval) and the node list and MAC list to empty lists.

When any node *A* receives a RREQ for which it is not the target, the node checks its local table of <initiator, id> values from recent requests it has received, to determine if it has already seen a request from this same Route Discovery. If it has, the node discards the packet, as in DSR. The node also checks whether the time interval in the request is valid: that time interval must not be too far in the future, and the key corresponding to it must not have been disclosed yet. If the time interval is not valid, the node discards the packet. Otherwise, the node modifies the request by appending its own address (*A*) to the node list in the request, replacing the hash chain field with H [A, *hash chain*], and appending a MAC of the entire REQUEST to the MAC list. The node uses the TESLA key K_{Ai} to compute the MAC, where *i* is the index for the time interval specified in the request. Finally, the node rebroadcasts the modified RREQ, as in DSR.

When the target node receives the RREQ, it checks the validity of the request by determining that the keys from the time interval specified have not been disclosed yet, and that the hash chain field is equal to:

$$\mathbf{H\,[\eta_n\,,\,H\,[\eta_{n-1}\,,\,H\,[\,\ldots\,,\,H\,[\eta_1\,,\,MAC_{KSD}\,(initiator,\,target,\,id,\,time\,interval)\,]\,\ldots\,]\,]\,]}$$

where η_i is the node address at position i of the node list in the request, and where n is the number of nodes in the node list. If the target node determines that the request is valid, it returns a RREP to the initiator, containing eight fields:

<ROUTE REPLY, target, initiator, time interval, node list, MAC list, target MAC, key list>

The target, initiator, time interval, node list, and MAC list fields are set to the corresponding values from the RREQ, the target MAC is set to a MAC computed on the preceding fields in the reply with the key KDS , and the key list is initialized to the empty list. The RREP is then returned to the initiator of the request along the source route obtained by reversing the sequence of hops in the node list of the request.

A node forwarding a RREP waits until it is able to disclose its key from the time interval specified, then it appends its key from that time interval to the key list field in the reply and forwards the packet according to the source route indicated in the packet. Waiting delays the return of the RREP but does not consume extra computational power.

When the initiator receives a RREP, it verifies that each key in the key list is valid, that the target MAC is valid, and that each MAC in the MAC list is valid. If all of these tests succeed, the node accepts the RREP; otherwise, it discards it.

In order to prevent the injection of invalid route errors into the network fabricated by any node other than the one on the sending end of the link specified in the error message, each node that encounters a broken link adds TESLA authentication information to the route error message, such that all nodes on the return path can authenticate the error. However, TESLA authentication is delayed, so all the nodes on the return path buffer the error but do not consider it until it is authenticated. Later, the node that encountered the broken link discloses the key and sends it over the return path, which enables nodes on that path to authenticate the buffered error messages.

ARIADNE is protected also from a flood of RREQ packets that could lead to the cache poisoning attack. Benign nodes can filter out forged or excessive RREQ packets using *Route Discovery chains*, a mechanism for authenticating route discovery, allowing each node to rate-limit discoveries initiated by any other node. The authors present two different approaches that can be found in [2].

ARIADNE is immune to the wormhole attack only in its advanced version: using an extension called TIK (TESLA with Instant Key disclosure) that requires tight clock synchronization between the nodes, it is possible to detect anomalies caused by a wormhole based on timing discrepancies.

3.3. ARAN

The ARAN secure routing protocol proposed by Dahill, Levine, Royer and Shields is conceived as an on-demand routing protocol that detects and protects against malicious actions carried out by third parties and peers in the ad hoc environment. ARAN introduces *authentication*, message *integrity* and *non-repudiation* as part of a minimal security policy for the ad hoc environment and consists of a preliminary certification process, a mandatory end-to-end authentication stage and an optional second stage that provides secure shortest paths.

ARAN requires the use of a trusted certificate server (T): before entering in the ad hoc network, each node has to request a certificate signed by T. The certificate contains the IP address of the node, its public key, a timestamp of when the certificate was created and a time at which the certificate expires along with the signature by T. All nodes are supposed to maintain fresh certificates with the trusted server and must know T's public key.

The goal of the first stage of the ARAN protocol is for the source to verify that the intended destination was reached. In this stage, the source trusts the destination to choose the return path. A source node, *A*, initiates the route discovery process to reach the destination *X* by broadcasting to its neighbors a route discovery packet called RDP:

$$\mathbf{[RDP;\ IP_X\ ;\ cert_A\ ;\ N_A\ ;\ t]K_{A-}}$$

The RDP includes a packet type identifier ("RDP"), the IP address of the destination (IP_X), *A*'s certificate ($cert_A$), a nonce N_A , and the current time t, all signed with A's private key. Each time A performs route discovery, it monotonically increases the nonce.

Each node records the neighbor from which it received the message. It then forwards the message to each of its neighbors, signing the contents of the message. This signature prevents spoofing attacks that may alter the route or form loops. Let *A*'s neighbor be *B*. It will broadcast the following message:

$$\mathbf{[[RDP;\ IP_X\ ;\ cert_A\ ;\ N_A\ ;\ t]K_{A-}\ ]K_{B-}\ ;\ cert_B}$$

Nodes do not forward messages for which they have already seen the (N_A ; IP_A) tuple. The IP address of A is contained in the certificate, and the monotonically increasing nonce facilitates easy storage of recently-received nonces.

Upon receiving the broadcast, *B*'s neighbor *C* validates the signature with the given certificate. *C* then rebroadcasts the RDP to its neighbors, first removing B's signature:

$$\mathbf{[[RDP;\ IP_X\ ;\ cert_A\ ;\ N_A\ ;\ t]K_{A-}\ ]K_{C-}\ ;\ cert_C}$$

Eventually, the message is received by the destination, *X*, who replies to the first RDP that it receives for a source and a given nonce. There is no guarantee that the first RDP received traveled along the shortest path from the source. The destination unicasts a Reply (REP) packet back along the reverse path to the source. Let the first node that receives the RDP sent by X be node *D*. *X* will send to *D* the following message:

$$\mathbf{[REP;\ IP_A\ ;\ cert_X\ ;\ N_A\ ;\ t]K_{X-}}$$

The REP includes a packet type identifier ("REP"), the IP address of A, the certificate belonging to *X*, the nonce and associated timestamp sent by *A*. Nodes that receive the REP forward the packet back to the predecessor from which they received the original RDP. All REPs are signed by the sender. Let *D*'s next hop to the source be node *C*. *D* will send to *C* the following message:

[[REP; IP_A ; $cert_X$; N_A ; t]K_{X-}]K_{D-} ; $cert_D$

C validates *D*'s signature, removes the signature, and then signs the contents of the message before unicasting the following RDP message to *B*:

[[REP; IP_A ; $cert_X$; N_A ; t]K_{X-}]K_{C-} ; $cert_C$

A node checks the signature of the previous hop as the REP is returned to the source. This avoids attacks where malicious nodes instantiate routes by impersonation and re-play of *X*'s message. When the source receives the REP, it verifies that the correct nonce was returned by the destination as well as the destination's signature. Only the destination can answer an RDP packet. Other nodes that already have paths to the destination cannot reply for the destination. While other protocols allow this networking optimization, ARAN removes several possible exploits and cuts down on the reply traffic received by the source by disabling this option.

The second stage of the ARAN protocol guarantees in a secure way that the path received by a source initiating a route discovery process is the shortest. Similarly to the first stage of the protocol, the source broadcasts a *Shortest Path Confirmation* (SPC) message to its neighbors: the SPC message is different from the RDP message only in two additional fields that provide the destination *X* certificate and the encryption of the entire message with *X*'s public key (which is a costly operation). The onion-like signing of messages combined with the encryption of the data prevents nodes in the middle from changing the path length because doing so would break the integrity of SPC the packet.

Also the route maintenance phase of the ARAN protocol is secured by digitally signing the route error packets. However it is extremely difficult to detect when error messages are *fabricated* for links that are truly active and not broken. Nevertheless, because messages are signed, malicious nodes cannot generate error messages for other nodes. The non-repudiation provided by the signed error message allows a node to be verified as the source of each error message that it sends.

As with any secure system based on cryptographic certificates, the key revocation issue has to be addressed in order to make sure that expired or revoked certificates do not allow the holder to access the network. In ARAN, when a certificate needs to be revoked, the trusted certificate server T sends a broadcast message to the ad hoc group that announces the revocation. Any node receiving this message re-broadcast it to its neighbors. Revocation notices need to be stored until the revoked certificate would have expired normally. Any neighbor of the node with the revoked certificate needs to reform routing as necessary to avoid transmission through the now un-trusted node. This method is not failsafe. In some cases, the un-trusted node that is having its certificate revoked may be the sole connection between two parts of the ad hoc network. In this case, the un-trusted node may not forward the notice of revocation for its certificate, resulting in a partition of the network, as nodes that have received the revocation notice will no longer forward messages through the un-trusted node, while all other nodes depend on it to reach the rest of the network. This only lasts as long as the un-trusted node's certificate would have otherwise been valid, or until the un-trusted node is no longer the sole connection between the two partitions. At the time

that the revoked certificate should have expired, the un-trusted node is unable to renew the certificate, and routing across that node ceases. Additionally, to detect this situation and to hasten the propagation of revocation notices, when a node meets a new neighbor, it can exchange a summary of its revocation notices with that neighbor; if these summaries do not match, the actual signed notices can be forwarded and re-broadcasted to restart propagation of the notice.

The ARAN protocol protects against exploits using *modification*, *fabrication* and *impersonation* but the use of asymmetric cryptography makes it a very costly protocol to use in terms of CPU and energy usage. Furthermore, ARAN is not immune to the *wormhole* attack

3.4. SEAD

Hu, Perrig and Johnson present a *proactive* secure routing protocol based on the Destination-Sequenced Distance Vector protocol (DSDV). In a proactive (or periodic) routing protocol nodes periodically exchange routing information with other nodes in attempt to have each node always know a current route to all destinations [7]. Specifically, SEAD is inspired by the DSDV-SQ version of the DSDV protocol. The DSDV-SQ version of the DSDV protocol has been shown to outperform other DSDV versions in previous ad hoc networks simulations [8, 9].

SEAD deals with attackers that *modify* routing information broadcasted during the update phase of the DSDV-SQ protocol: in particular, routing can be disrupted if the attacker modifies the sequence number and the metric field of a routing table update message. *Replay attacks* are also taken into account.

In order to secure the DSDV-SQ routing protocol, SEAD makes use of efficient *one-way hash chains* rather than relaying on expensive asymmetric cryptography operations. However, like the other secure protocols presented in this chapter, SEAD assumes some mechanism for a node to distribute an authentic element of the hash chain that can be used to authenticate all the other elements of the chain. As a traditional approach, the authors suggest to ensure the key distribution relaying on a trusted entity that signs public key certificates for each node; each node can then use its public key to sign a hash chain element and distribute it.

The basic idea of SEAD is to authenticate the sequence number and metric of a routing table update message using hash chains elements. In addition, the receiver of SEAD routing information also authenticates the sender, ensuring that the routing information originates form the correct node.

To create a one-way hash chain, a node chooses a random initial value $x \in \{0,1\}^{\rho}$, where ρ is the length in bits of the output of the hash function, 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. As an 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 hi.

Each node uses a specific authentic (i.e. signed) 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 provides authentication for the lower bound on the metric in other routing updates for that node. The use of a hash value corresponding to the

sequence number and metric in a routing update entry prevents any node from advertising a route to some destination claiming a greater sequence number than that destination's own current sequence number. Likewise, a node can not advertise a route better than those for which it has received an advertisement, since the metric in an existing route cannot be decreased due to the on-way nature of the hash chain.

When a node receives a routing update, it checks the authenticity of the information for each entry in the update using the destination address, the sequence number and the metric of the received entry, together with the latest prior *authentic* hash value received from that destination's hash chain. Hashing the received elements the correct number of times (according to the prior authentic hash value) assures the authenticity of the received information if the calculated hash value and the authentic hash value match.

The source of each routing update message in SEAD must also be authenticated, since otherwise, an attacker may be able to create routing loops through the *impersonation* attack. The authors propose two different approaches to provide node authentication: the first is based on a broadcast authentication mechanism such as TESLA, the second is based on the use of Message Authentication Codes, assuming a shared secret key between each couple of nodes in the network.

SEAD does not cope with *wormhole* attacks though the authors propose, as in the ARIADNE protocol, to use the TIK protocol to detect the threat.

3.5. Notes on the wormhole attack

The wormhole attack is a severe threat against ad hoc routing protocols that is particularly challenging to detect and prevent. In a wormhole attack a malicious node can record packets (or bits) at one location in the network and tunnel them to another location through a private network shared with a colluding malicious node. Most existing ad hoc routing protocols, without some mechanism to defend them against the wormhole attack, would be unable to find consistent routes to any destination, severely disrupting communication.

A dangerous threat can be perpetrated if a wormhole attacker tunnels all packets through the wormhole honestly and reliably since no harm seems to be done: the attacker actually seems to provide a useful service in connecting the network more efficiently. However, when an attacker forwards only routing control messages and not data packets, communication may be severely damaged. As an example, when used against an on demand routing protocol such as DSR, a powerful application of the wormhole attack can be mounted by tunneling each RREQ message directly to the destination target node of the request. This attack prevents routes more than two hops long from being discovered because RREP messages would arrive to the source faster than any other replies or, worse, RREQ messages arriving from other nodes next to the destination than the attacker would be discarded since already seen.

Hu, Perrig and Johnson propose an approach to detect a wormhole based on *packet leashes* [10]. 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 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. Given the precise time synchronization required by temporal leashes, the authors propose efficient broadcast authenticators based on symmetric primitives. In particular they extend the TESLA broadcast authentication protocol to allow the disclosure of the authentication key within the packet that is authenticated.

Geographical leashes are based on location information and loosely synchronized clocks. If the clocks of the sender and the receiver are synchronized within a certain threshold and the velocity of any node is bounded, the receiver can compute an upper bound on the distance between the sender and itself and use it to detect anomalies in the traffic flow. In certain circumstances however, bounding the distance between the sender and the receiver cannot prevent wormhole attacks: 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 the receiver. To overcome this problem, in a variation of the geographical leashes the receiver verifies that every possible location of the sender can reach every possible location of the receiver based on a radio propagation model implemented in every node.

In some special cases, wormholes can also be detected through techniques that don't require precise time synchronization nor location information. As an example, it would be sufficient to modify the routing protocol used to discover the path to a destination so that it could handle multiple routes: a verification mechanism would then detect anomalies when comparing the metric (e.g. number of hops) associated to each route. Any node advertising a path to a destination with a metric considerably lower than all the others could raise the suspect of a wormhole.

Furthermore, if the wormhole attack is performed only on routing information while dropping data packets, other mechanisms can be used to detect this misbehavior. When a node doesn't correctly participate to the network operation by not executing a particular function (e.g. packet forwarding) a collaborative monitoring technique can detect and gradually isolate misbehaving nodes. Lack of cooperation and security mechanism used to enforce node cooperation to the network operation is the subject of the next section.

4. Selfishness and Cooperation Enforcement

Selfishness is a new type of misbehavior that is inherent to ad hoc networks and cooperation enforcement is the countermeasure against selfishness. A selfish node does not directly intend to damage other nodes with active attacks (mainly because performing active attacks can be very expensive in terms of energy consumption) but it simply does not contribute in the network operation, saving battery life for its own communications. Selfishness can cause serious damage in terms of global network throughput and delay as shown by a simulation study on the impact of selfish behavior on the DSR routing protocol [11]. The node selfishness problem has only

recently been addressed by the research community, and still very few cooperation enforcement mechanisms are proposed to combat such misbehavior. Current cooperation enforcement proposals for MANET fall in two categories: currency-based solutions whereby some form of digital cash is used as an incentive for cooperation and monitoring solutions based on the principle that misbehaving nodes will be detected through the shared observations of a majority of legitimate nodes. The most significant proposals in each category are outlined in the sequel of this section.

4.1. Nuglets

In [14], Buttyan and Hubaux present two important issues targeted specifically at the ad hoc networking environment: first, end-users must be given some incentive to contribute in the network operation (especially to relay packets belonging to other nodes); second, end-users must be discouraged from overloading the network. The solution consists of a virtual currency call Nuglet used in every transaction. Two different models are described: the Packet Purse Model and the Packet Trade Model. In the Packet Purse Model each packet is loaded with nuglets by the source and each forwarding host takes out nuglets for its forwarding service. The advantage of this approach is that it discourages users from flooding the network but the drawback is that the source needs to know exactly how many nuglets it has to include in the packet it sends. In the Packet Trade Model each packet is traded for nuglets by the intermediate nodes: each intermediate node buys the packet from the previous node on the path. Thus, the destination has to pay for the packet. The direct advantage of this approach is that the source does not need to know how many nuglets need to be loaded into the packet. On the other hand, since the packet generation is not charged, malicious flooding of the network cannot be prevented. There are some further issues that have to be solved: concerning the Packet Purse Model, the intermediate nodes are able to take out more nuglets than they are supposed to; concerning the Packet Trade Model, the intermediate nodes are able to deny the forwarding service after taking out nuglets from a packet.

4.2. CONFIDANT

Buchegger and Le Boudec proposed a technique called CONFIDANT (Cooperation Of Nodes, Fairness In Dynamic Ad-hoc NeTworks) [15,16] aiming at detecting malicious nodes by means of combined monitoring and reporting and establishing routes by avoiding misbehaving nodes. CONFIDANT is designed as an extension to a routing protocol such as DSR. CONFIDANT components in each node include a network monitor, 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 the behavior of the local node 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.

The dynamic behavior of CONFIDANT is as follows. Nodes monitor their neighbors and change the reputation accordingly. If they have a 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 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 misbehaved node in the form of excluding routes containing the misbehaved node, re-ranking paths in the path cache, reciprocating by non-cooperation, and forwarding an ALARM about the node.

The first version of CONFIDANT was, despite the filtering of ALARM messages in the trust manager, vulnerable to concerted efforts of spreading wrong accusations. In a recent enhancement of the protocol, this problem has been addressed by the use of Bayesian statistics for classification and the exclusion of liars.

Simulations with nodes that do not participate in the forwarding function have shown that CONFIDANT can cope well, even if half of the network population acts maliciously. Further simulations concerning the effect of second-hand information and slander have shown that slander can effectively be prevented while still retaining a significant detection speed-up over using merely first-hand information.

The limitations of CONFIDANT lie in the assumptions for detection-based reputation systems. Events have to be observable and classifiable for detection, and reputation can only be meaningful if the identity of each node is persistent, otherwise it is vulnerable to spoofing attacks.

4.3. CORE

The security scheme proposed by Michiardi and Molva [18, 19], stimulates node cooperation 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 using a sophisticated reputation mechanism that differentiates between subjective reputation (observations), indirect reputation (positive reports by others), and functional reputation (task-specific behavior), which are weighted for a combined reputation value. 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.

As for the other security mechanism based on reputation the CORE mechanism suffers from the spoofing attack: misbehaving nodes are not prevented from changing their network identity allowing the attacker to elude the reputation system. Furthermore, no simulation results prove the robustness of the protocol even if the authors propose an original approach based on game theory in order to come up with a formal assessment of the security properties of CORE.

4.4. Token-based cooperation enforcement

In [20] Yang, Meng, Lu suggest a mechanism whereby each node of the ad hoc network is required to hold a token in order to participate in the network operations. Tokens are granted to a node collaboratively by its neighbors based on the monitoring of the node's contribution to packet forwarding and routing operations. Upon expiration of the token, each node renews its token through a token renewal exchange with its neighbors: the duration of a token's validity is based on the duration of the node's correct behavior as monitored by the neighbors granting/renewing the token. This mechanism typically allows a well-behaved node to accumulate credit and to renew its token less frequently as time evolves.

The token-based cooperation enforcement mechanism includes four interacting components: *neighbor verification* through which the local node verifies whether neighboring nodes are legitimate, *neighbor monitoring* that allows the local node to monitor the behavior of each node in the network and to detect attacks from malicious nodes, *intrusion reaction* that assures the generation of network alerts and the isolation of attackers, and *security enhanced routing protocol* that consists of the ad hoc routing protocol including security extensions.

A valid token is constructed using a group signature whereby a mechanism based on polynomial secret sharing [25] assures that at least k neighbors agree to issue or renew the token. The key setup complexity of polynomial secret sharing and the requirement for at least k nodes to sign each token both are incompatible with high mobility and call for a rather large and dense ad hoc network. Furthermore the duration of a token's validity increases proportionally with the duration of the node's correct behavior as monitored by its neighbors; this feature again calls for low mobility. The token-based cooperation enforcement mechanism is thus suitable for ad hoc networks where node mobility is low. Spoofing attacks through which a node can request more than one token claiming different identities, are not taken into account

by the proposal even if the authors suggest that MAC addresses can be sufficient for node authentication purposes.

5. Authentication and Key Management

Authentication of peer entities involved in ad hoc routing and the integrity verification of routing exchanges are the two essential building blocks of secure routing. Both entity authentication and message integrity call on the other hand for a key management mechanism to provide parties involved in authentication and integrity verification with proper keying material. Key management approaches suggested by current secure routing proposals fall in two categories:

- manual configuration of symmetric (secret) keys: the pair-wise secret keys can serve as key encryption keys in a point-to-point key exchange protocol to establish session keys used for authentication and message integrity between communicating nodes. If some dedicated infrastructure including a key server can be afforded, automatic distribution of session keys with a key distribution protocol like Kerberos can also be envisioned.
- public-key based scheme: each node possesses a pair of public and private keys based on an asymmetric algorithm like RSA. Based on this keypair each node can perform authentication and message integrity operations or further exchange pair-wise symmetric keys used for efficient authentication and encryption operations.

Secure routing proposals like SRP assume manual configuration of secure associations based on shared secret keys. Most of other proposals such as Ariadne rely on a public-key based scheme whereby a well known trusted third party (TTP) issues public key certificates used for authentication. The requirement for such a public-key infrastructure does not necessarily imply a managed ad hoc network environment and an open environment can be targeted as well. Indeed, it is not necessary for the mobile nodes that form the ad hoc network to be managed by the public-key certification authority. However, the bootstrap phase requires an external infrastructure, which has to be available also during the lifetime of the ad hoc network to provide revocation services for certificates that have expired or that have been explicitly revoked.

Two interesting proposals presented in the next section tackle the complexity of public-key infrastructures in the ad hoc network environment through self-organization: public-key management based on the concept of web of trust akin to Pretty Good Privacy (PGP) and a public-key certification mechanism based on polynomial secret sharing.

5.1. Self-Organized Public-Key Management based on PGP

Capkun, Buttyan and Hubaux propose a fully self-organized public key management system that can be used to support security of ad hoc network routing

protocols [21]. The suggested approach is similar to PGP [22] 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 on-line 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.

The success of this approach very much depends on the construction of the local certificate repositories and on the characteristics of the certificate graphs. The vertices of a certificate graph represent public-keys of the users and the edges represent public-key certificates issued by the users. The authors investigate 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 mobile node depends on the connectivity of the node and its neighbors in the certificate graph.

More precisely, each node stores in its local repository several directed and mutually disjoint paths of certificates. Each path begins at the node itself, and the certificates are added to the path such that a new certificate is chosen among the certificates connected to the last node on the path (initially the node that stores the certificates), such that the new certificate leads to the node that has the highest number of certificates connected to it (i.e., the highest vertex degree). The authors call this algorithm the *Maximum Degree Algorithm*, as the local repository construction criterion is the degree of the vertices in a certificate graph.

In a more sophisticated extension called the *Shortcut Hunter Algorithm*, certificates are stored into the local repositories based on the number of the shortcut certificates connected to the users. The shortcut certificate is a certificate that, when removed from the graph makes the shortest path between two users previously connected by this certificate strictly larger than two.

When verifying a certificate chain, the node must trust the issuer of the certificates in the chain for correctly checking that the public key in the certificate indeed belongs to the node identification (ID) named in the certificate. When certificates are issued by the mobile nodes of an ad hoc network instead of trusted authorities, this assumption becomes unrealistic. In addition, there may be malicious nodes who issue false certificates. In order to alleviate these problems, the authors propose the use of authentication metrics [23]: it is not enough to verify a node ID key binding via a single chain of certificates. The authentication metric is a function that accepts two keys (the verifier and the verified node) and a certificate graph and returns a numeric value corresponding to the degree of authenticity of the key that has to be verified: one example of authentication metric is the number of disjoint chains of certificates between two nodes in a certificate graph.

The authors emphasize that before being able to perform key authentication, each node must first build its local certificate repository, which is a complex operation. However this initialization phase must be performed rarely and once the certificate repositories have been built, then any node can perform key authentication using only local information and the information provided by the targeted node. It should also be noted that local repositories become obsolete if a large number of certificate are

revoked, as then the certificate chains are no longer valid; the same comment applies in the case when the certificate graph changes significantly. Furthermore, PGP-like schemes are more suitable for small communities because that the authenticity of a key can be assured with a higher degree of trustiness. The authors propose the use of authentication metrics to alleviate this problem: this approach however provides only probabilistic guarantees and is dependent on the characteristics of the certificate graph on which it operates. The authors also carried out a simulation study showing that for the certificate graphs that are likely to emerge in self-organized systems, the proposed approach yields good performances both in terms of the size of the local repository stored in each node and scalability.

5.2. Authentication based on polynomial secret sharing

In [24] Luo and Lu present an authentication service whereby the public-key certificate of each node is cooperatively generated by a set of neighbors based on the behavior of the node as monitored by the neighbors. Using a group signature mechanism based on polynomial secret sharing, the secret digital signature key used to generate public-key certificates is distributed among several nodes. Certification services like issuing, renewal and revocation of certificates thus are distributed among the nodes: a single node holds just a share of the complete certificate signature key. The authors propose a *localized trust model* to characterize the localized nature of security concerns in large ad hoc wireless networks. When applying such trust model, an entity is trusted if any k trusted entities claim so: these k trusted entities are typically the neighboring nodes of the entity. A locally trusted entity is globally accepted and a locally distrusted entity is regarded untrustworthy all over the network.

In the suggested security architecture, each node carries a certificate signed by the shared certificate signing key SK, while the corresponding public key PK is assumed to be well-known by all the nodes of the network, so that certificates are globally verifiable. Nodes without valid certificates will be isolated, that is, their packets will not be forwarded by the network. Essentially, any node without a valid certificate is considered a potential intruder. When a mobile node moves to a new location, it exchanges certificates with its new neighbors and goes through mutual authentication process to build trust relationships. Neighboring nodes with such trust relationship help each other to forward and route packets. They also monitor each other to detect possible attacks and break-ins. Specific monitoring algorithms and mechanisms are left to each individual node's choice. When a node requests a signed certificate from a coalition of k nodes, each of the latter checks its records about the requesting node. If the requestor is recorded as a legitimate node, a partial certificate is computed by applying the local node's share of SK and returned to the requestor. Upon collecting k partial certificates, the requesting node combines them to generate the complete certificate of its public-key as if issued by a centralized certification authority.

The multiple signature scheme used to build the certificate is based on a k-threshold polynomial secret sharing mechanism. This technique requires a bootstrapping phase where a "dealer" has to privately send each node its share of the secret signature key SK. The authors propose a scalable initialization mechanism called "self-initialization" whereby the dealer only has to initialize the very first k

nodes, regardless of the global network span. The initialized nodes collaboratively initialize other nodes: repeating this procedure, the network progressively self-initializes itself. The same mechanism is applied when new nodes join the network.

Certificate revocation is also handled by the proposed architecture and an original approach to handle roaming adversaries is presented in order to prevent a misbehaving node that moves to a new location from getting a valid certificate. Roaming nodes are defeated with the flooding of "accusation" messages that travel in the network and inform distant nodes about the behavior of a suspect node.

The main drawback of the proposed architecture is the requirement for a trusted dealer that initializes the very first *k* nodes of a coalition to the choice of the system-wide parameter *k*. To cope with the first problem, the authors propose to use a distributed RSA key pair generation [25] for the very first *k* nodes. The other major limitation of the scheme is the strong assumption that every node of the network has at least *k* trusted neighbors. Moreover, the authors assume that any new node that joins the system already has an initial certificate issued by an offline authority or by a coalition of *k* neighbors.

6. MANET and Data Link Layer Security

Various security mechanisms have been proposed as part of 802.11[26] and Bluetooth [27] specifications. While the robustness of these mechanisms has often been argued [29], the main question is the relevance of security mechanisms implemented in the data link layer with respect to the requirements of MANET. This question deserves careful analysis in the light of requirements raised by the two different environments in which these mechanisms can potentially be deployed:

1. wireless extension of a wired infrastructure as the original target of 802.11 and Bluetooth security mechanisms,
2. wireless ad hoc networks with no infrastructure.

In case of 1 the main requirement for data link layer security mechanisms is the need to cope with the lack of physical security on the wireless segments of the communication infrastructure. Data link layer security is then perfectly justified as a means of building a "wired equivalent" security as stated by the objectives of Wired Equivalent Privacy (WEP) of 802.11. Data link layer mechanisms like the ones provided by 802.11 and Bluetooth basically serve for access control and privacy enhancements to cope with the vulnerabilities of radio communication links. However, data link layer security performed at each hop cannot meet the end-to-end security requirements of applications neither on wireless links protected by 802.11 or Bluetooth nor on physically protected wired links.

In case of wireless ad hoc networks as defined in 2 there are two possible scenarios:

- managed environments whereby the nodes of the ad hoc network are controlled by an organization and can thus be trusted based on authentication,

- open environments with no a priori organization among network nodes.

The managed environment raises requirements similar to the ones of 1. Data link layer security is justified in this case by the need to establish a trusted infrastructure based on logical security means. If the integrity of higher layer functions implemented by the nodes of a managed environment can be assured (i.e. using tamper-proof hardware) then data link layer security can even meet the security requirements raised by higher layers including the routing protocol and the applications.

Open environments on the other hand offer no trust among the nodes and across communication layers. In this case trust in higher layers like routing or application protocols cannot be based on data link layer security mechanisms. The only relevant use of the latter appears to be ad hoc routing security proposals whereby the data link layer security can provide node-to-node authentication and data integrity as required by the routing layer. Moreover the main impediment to the deployment of existing data link layer security solutions (802.11 and Bluetooth) would be the lack of support for automated key management which is mandatory in open environments whereby manual key installation is not suitable.

7. Conclusion

Security of ad hoc networks has recently gained momentum in the research community. Due to the open nature of ad hoc networks and their inherent lack of infrastructure, security exposures can be an impediment to basic network operation and the countermeasures should be included in network functions from the early stages of their design. Security solutions for MANET have to cope with a challenging environment including scarce energy and computational resources and lack of persistent structure to rely on for building trust.

The solutions presented in this article only cover a subset of all threats and are far from providing a comprehensive answer to the security problem in ad hoc networks. They often address isolated issues away from a global approach to security: for instance, secure routing proposals do not take into account lack of cooperation and do not include cooperation enforcement mechanisms. Most routing security solutions also make unrealistic assumptions about the availability of key management infrastructures that are in contrast with the very nature of ad hoc networks. As the technology for ad hoc wireless networks gains maturity, comprehensive security solutions based on realistic trust models and addressing all prevalent issues like routing, key management and cooperation enforcement are expected to appear.

8. Bibliography

[1] P. Papadimitratos, Z. Haas, Secure Routing for Mobile Ad Hoc Networks, in proceedings of CNDS 2002.

[2] Y-C Hu, A. Perrig, D. B. Johnson, Ariadne : A secure On-Demand Routing Protocol for Ad Hoc Networks, in proceedings of MOBICOM 2002.

[3] A. Perrig, R. Canetti, D. Song, J.D. Tygar, Efficient and secure source authentication for multicast, in proceedings of NDSS 2001.

[4] A. Perrig, R. Canetti, J.D. Tygar, D. Song, Efficient authentication and signing of multicast streams over lossy channels, in IEEE Symposium on Security and Privacy, 2000.

[5] B. Dahill, B. N. Levine, E. Royer, C. Shields, ARAN: A secure Routing Protocol for Ad Hoc Networks, UMass Tech Report 02-32, 2002.

[6] Y-C Hu, D. B. Johnson, A. Perrig, SEAD: Secure Efficient Distance Vector Routing for Mobile Wireless Ad Hoc Networks, in the Fourth IEEE Workshop on Mobile Computing Systems and Applications.

[7] C. E. Perkins, P. Bhagwat, Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers, in proceedings of SIGCOMM 1994.

[8] J. Broch, D. A. Maltz, D. B. Johnson, Y-C Hu, J. G. Jetcheva, A performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols, in proceedings of MOBICOM 1998.

[9] P. Johansson, T. Larsson, N. Hedman, B. Mielczarek, M. Degermark, Scenario-based Performance Analysis of Routing Protocols for Mobile Ad Hoc Networks, in proceedings of MOBICOM 1999.

[10] A. Perrig, Y-C Hu, D. B. Johnson, Wormhole Protection in Wireless Ad Hoc Networks, Technical Report TR01-384, Dep. Of Computer Science, Rice University.

[11] P. Michiardi, R. Molva, Simulation-based Analysis of Security Exposures in Mobile Ad Hoc Networks, in proceedings of European Wireless Conference, 2002.

[12] D. B. Johnson, D. A. Maltz, Dynamic Source Routing in Ad Hoc Wireless Networks, Mobile Computing, edited by Tomasz Imielinski and Hank Korth, Chapter 5, pages 153-181, Kluwer Academic Publishers, 1996.

[13] Charles Perkins, Ad hoc On Demand Distance Vector (AODV) Routing, Internet draft, draft-ietf-manet-aodv-00.txt.

[14] L. Buttyan, J.-P. Hubaux, Nuglets: a virtual currency to stimulate cooperation in self-organized ad hoc networks, Technical Report DSC/2001/001, Swiss Federal Institute of Technology -- Lausanne, 2001.

[15] S. Buchegger, J.-Y. Le Boudec, Nodes Bearing Grudges: Towards Routing Security, Fairness, and Robustness in Mobile Ad Hoc Networks, in proceedings of the 10th Euromicro Workshop on Parallel, Distributed and Network-based Processing.

[16] S. Buchegger, J.-Y. Le Boudec, Performance Analysis of the CONFIDANT Protocol, in proceedings of MobiHoc 2002.

[17] S. Marti, T. Giuli, K. Lai, and M. Baker, Mitigating routing misbehavior in mobile ad hoc networks, in proceedings of MOBICOM 2000.

[18] P. Michiardi, R. Molva, Core: A COllaborative REputation mechanism to enforce node cooperation in Mobile Ad Hoc Networks, IFIP - Communication and Multimedia Security Conference 2002.

[19] P. Michiardi, R. Molva, Game Theoretic Analysis of Security in Mobile Ad Hoc Networks, Institut Eurecom Research Report RR-02-070 - April 2002.

[20] H. Yang, X. Meng, S. Lu, Self-Organized Network-Layer Security in Mobile Ad Hoc Networks.

[21] S. Capkun, L. Buttyan and J-P Hubaux, Self-Organized Public-Key Management for Mobile Ad Hoc Networks, in ACM International Workshop on Wireless Security, WiSe 2002.

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

[23] M. Reiter, S. Stybblebine, Authentication metric analysis and design, ACM Transactions on Information and System Security, 1999.

[24] H. Luo, S. Lu, Ubiquitous and Robust Authenticaion Services for Ad Hoc Wireless Networks, UCLA-CSD-TR-200030.

[25] A. Shamir, How to share a secret, Communications of ACM 1979.

[26] IEEE 802.11b-1999 Supplement to 802.11-1999,Wireless LAN MAC and PHY specifications: Higher speed Physical Layer (PHY) extension in the 2.4 GHz band

[27] "Specification of the Bluetooth System", Bluetooth Special Interest Group, Version 1.1, February 22, 2001, http://www.bluetooth.com/pdf/Bluetooth_11_Specifications_Book.pdf

[28] "Applied Cryptography", Bruce Schneier, John Wiley & Sons, 1996

[29] "Using the Fluhrer, Mantin, and Shamir Attack to Break WEP", Stubblefield, Loannidis, and Rubin, AT&T Labs Technical Report 2001

[30] "Providing robust and ubiquitous security support for manet", J. Kong, P. Zerfos, H. Luo, S. Lu, and L. Zhang, In Proc. IEEE ICNP, 2001.