Secure Dynamic Source Routing

Frank Kargl, Alfred Geiß, Stefan Schlott, Michael Weber
University of Ulm, Germany
{givenname.surname}@informatik.uni-ulm.de

***Abstract*— In this paper we present the Secure Dynamic Source Routing protocol for Mobile Ad hoc Networks that prevents a lot of potential attacks to these kind of networks. We also present a number of similar protocols and compare the different approaches. After a detailed description of SDSR, we show that the stated security goals are met using the BAN logic formalism.**

I. Introduction

Mobile Ad hoc Networks (MANETs) represent an uncommon way of building and organizing a network of mobile nodes without any kind of infrastructure. When speaking of MANETs in this paper we refer to multi-hop ad hoc networks where intermediary nodes relay traffic for others.

In order to organize such networks you need a routing protocol which is able to discover the network topology and build routes in a very dynamic environment where the position of nodes and thus the availability of links between them changes rapidly.

A large number of such routing protocols has been proposed in the recent years, like Ad hoc On demand Distance Vector Routing (AODV) or Dynamic Source Routing (DSR) to name only the most prominent representatives.

A. DSR

As our work is based on the DSR protocol as a basis, we describe the operation of DSR a little more detailed. DSR is a proactive (or on-demand) routing protocol which searches for routes between two nodes only when there are actual packets waiting to be transmitted from source S to destination D.

It therefore initiates a route discovery, flooding the network with a route request. During this flooding process the nodes create a list of traversed nodes in the packets. When a route request packet reaches the destination, this list is considered a valid route from the source to the destination[1] and transmitted to the source in a route reply packet. Route replies need not to be flooded again, as they can be transfered along the reverse route[2]. DSR is a source routing protocol, i.e. the complete route is given in the header of each packet. Intermediary nodes use this information for determining the next hop.

The DSR specification contains a lot of additional functions and details, esp. a number of optimizations to enhance performance. As some of these optimizations can oppose security, we discuss these in section V.

DSR does not address security, the draft [1] rather states:

> "This document does not specifically address security concerns. This document does assume that all nodes participating in the DSR protocol do so in good faith and without malicious intent to corrupt the routing ability of the network."

There are a number of publications that use DSR as starting point for developing a secure MANET routing protocol. These are described in section II.

B. Security in MANETs

MANETs share the basic security goals with most other networks. The need for Confidentiality, Authenticity, Integrity, Availability, Non-Repudiation and Access Control is the same as in other types of networks and is mainly determined by the importance and sensitivity of applications used or data transmitted.

Nevertheless as ad hoc networks can not assume a central administration or coordination, as the participants of the network and their relative position changes quickly and as the network is created in a cooperative effort, these goals are harder to achieve than in conventional networks.

[1]And vice versa in the case of bidirectional links. In this paper we assume bidirectional links, as authentication of nodes and integrity checking of routes is extremely hard to achieve otherwise.

[2]Again this holds true only in the case of bidirectional links

The special security problems faced in MANETs include

- **Selfish Behavior:** resources[3] are very restricted in most mobile devices. So nodes may decide not to invest local resources for relaying packets but instead use the MANET only for transmitting own traffic. Preventing relaying of foreign packets may be done in a number of ways:
  - Simply don't forward packets that a node receive for other nodes.
  - Prevent own node to be chosen in routes. This can be accomplished by refusing to forward route requests or by altering route requests/replies so that routes including this node get very long and unattractive.

  Studies have shown that selfishness can severely affect the service quality of a MANET [2], [3].
- **Malicious Behavior:** for different reasons there may be nodes in a MANET that want to actively attack the network. As all nodes are part of the routing infrastructure, these attacks can be done easily and do a lot of damage. Different forms of attacks include
  - Denial of Service: attackers may try to interfere on the radio level, flood the network with data, or disturb the topology building process of the routing protocol. This can result in routing loops, black holes etc.
  - Reroute Traffic: attackers may forge routing information and manipulate routes in order to reroute packets through a specific node in order to gain extended access to packets.
- **Information leakage:** MANETs may contain a lot of valuable information that an attacker might want to access. This contains data transmitted in the network as well as "metadata" like node positions to create location profiles of users.

All these problems and attacks might negatively affect an ad hoc network – either in terms of reliability or bandwidth, or with respect to the trust that a user might put in the safety of the network. So security mechanisms that reach the goals outlined above are clearly mandatory for MANETs.

Current security related research activities for MANETs roughly divide into three categories:

1) **Identification of nodes:** nodes in MANET need to be identifiable so that nodes can not spoof each other's identity etc. In order to avoid location profiling, the use of pseudonyms or similar mechanisms to protect the privacy of users might be useful. Identities are often realized using cryptographic keys that offer additional possibilities like encryption of traffic etc.
   In literature, two major approaches can be observed. [4] suggests the use of threshold cryptography schemes to create a distributed certification authority. Threshold cryptography shares a secret among n participants in a way so that k of n participants can reconstruct the secret. Variations of that scheme allow the distributed creation of a public key pair and a distributed signing process.
   Another approach turns down the use of a CA completely. The authors of [5] suggest a scheme which uses a web of trust, similar to the encryption tool PGP. Every participant creates a public key pair of his own. When a participant is sure about the identity of another node, he signs the correspondent public key, certifying its identity. By following those "a certifies identity of b" links, the identity of a new node may be verified.
2) **Securing routing protocol against manipulations:** using various cryptographic mechanisms, manipulations during the routing process should be prevented. As the rest of the paper focuses on this aspect, we do not go into details here.
3) **Preventing selfish behavior:** there are different ideas how this can be accomplished: *Motivation-based approaches* try to motivate network users to actively participate in the MANET. A typical system representing this approach is Nuglets by Hubeaux et al. [6], [7]. The authors suggest to introduce a virtual currency called Nuglets that is earned by relaying foreign traffic and spent by sending own traffic. The major drawback of this approach is the demand for trusted hardware to secure the currency. There are arguments that tamper-resistant devices in general might be next to impossible to be realized [8], [9]. A similar approach without the need of tamper-proof hardware has been suggested by Zhong et al. in [10].
   Most other approaches try to *detect and exclude* selfish nodes. One example is the work of Marti, Giuli, Lai, and Baker in [11]. Their system uses a watchdog that monitors the neighboring

[3]like battery power, network bandwidth, memory, or computing power

nodes to check if they actually relay the data the way they should do. Then a component called pathrater will try to prevent paths which contain such misbehaving nodes. In [12], [13], the authors describe a distributed intrusion detection system (IDS) for MANETs that consists of the local components "data collection", "detection" and "response", and of the global components "cooperative detection" and "global response". Another system is the "Collaborative Reputation Mechanism" or CORE [14], [15]. It is similar to the distributed IDS by Zhang et al. and consists of local observations that are combined and distributed to calculate a reputation value for each node. Based on this reputation, nodes are allowed to participate in the network or are excluded. Yet a similar approach is conducted by Buchegger et al. with their system called CONFIDANT [16], [17].
Finally we have suggested a *Mobile Intrusion Detection System (MobIDS)*, that focuses especially on integration with other mechanisms and on refinement of sensors for detecting selfish nodes.

These different areas are closely related and there exist a large number of dependencies in between. Many authentication systems for ad-hoc networks assume working routes so they can reach other nodes. On the other hand many secure routing protocols assume pre-established shared keys and authenticated nodes. This clearly creates a conflict.

Furthermore many intrusion detection systems that try to detect selfish nodes in ad hoc networks assume that they know the network topology in order to detect selfish nodes which do not forward traffic correctly. So the IDS has to be integrated in the routing protocol.

These are just two examples that illustrate the need for a comprehensive security framework that combines the different security mechanisms in a consistent manner. The work presented here is part of a larger framework called Security Architecture for Mobile Ad hoc Networks (SAM) [18] that consists of the components MANET-IDs (for identification of nodes), SDSR (the routing protocol presented here), and MobIDS (a mobile intrusion detection system to find selfish nodes). A detailed description can be found in [19].

II. Routing Security in MANETs

One important piece in SAM is the security of the routing protocol itself. Attackers should not be able to disturb the routing process, so that they can create false routes that lead to denial of service or suboptimal routes. We have developed a secure variant of the DSR protocol called *Secure Dynamic Source Routing (SDSR)* with the following goals:

1) *Ensuring route integrity:* no node should be able to alter or modify the routing process, so denial of service attacks, creation of black holes that absorb traffic, or any other form of destructive attack will become impossible.
2) *Ensuring route freshness:* this will prevent nodes from replaying old and potentially stale routing packets that may also lead to wrong routes.
3) *Authentication of participating nodes:* after a route has been established, the source and destination node should be sure that the nodes forming the route are actually authentic and that no node is spoofing another node's identity.
4) *Exchange of session keys:* in order to protect data traffic, source and destination should securely exchange a session key that can be used for encryption of data packets. Additionally the intrusion detection system MobIDS demands session keys exchanged between source and destination and all intermediary nodes forming a route.
5) *Low Overhead:* Since SAM and SDSR are expected to work on small devices like PDAs or cellphones, the overhead generated by these mechanisms must remain small. Thus computational intensive cryptography, unnecessary traffic or large memory requirements should be avoided where possible.

Ensuring route integrity also prevents some forms of selfishness, since nodes are unable to lengthen routes leading through themselves, which may lead to other routes being chosen. The routing protocol does not prevent nodes from just not participating in the routing process. We think that this goal can be better reached by using an intrusion detection system like MobIDS outlined above.

Next we will describe some of the earlier work on secure MANET routing protocols and compare them to our solution. For a complete and detailed discussion of these approaches, see [19].

The *Secure Ad hoc On-Demand Distance Vector* routing protocol (SAODV) [20], [21], [22] is an extension to AODV [23]. SAODV assumes an pre-established public key infrastructure that distributes

Function	SAODV	Ariadne	ARAN	SRP	SDSR
Key Distribution	presumed	presumed	integrated	presumed	integrated
Node Authentication	only end nodes	all	all	only end nodes	all
Secure route in request	yes (ext. possible)	yes	yes (ext. possible)	no	yes
Secure route in reply	yes	yes	yes	yes	yes
Assure route freshness	yes	yes	yes	yes	yes
Exchange symmetric keys	no	no	no	no	yes
Use of cached routes	yes	no	no	no	no
Performance	⊕	○	⊖	⊕⊕	○
add. requisites	none	synch. clocks	synch. clocks	none	none

TABLE I

Comparison of different secure MANET routing protocols

signed public keys to all participants. Participants can now encrypt traffic to other nodes or verify signatures. All static parts of a route request or reply packet are protected from alteration by a signature using the private key of the sender. The only mutable part, the hop count, also needs protection because otherwise nodes may be able to shorten or lengthen routes. This protection is done using so called hash-chains. The hash chain is computed by the sender of the packet using a known hash function and a random number (seed) as start value. The result (so called top hash) and the seed are stored in the packet. Every station that forwards the packet, increments the hop count and also computes a new hash by applying the hash function to the hash value in the packet. Each station can then verify if hop count and position in the hashchain correspond. No station can decrement the hop count and shorten the route and thus attract traffic. Nevertheless increasing the hop count by an arbitrary number and thus rejecting unwanted traffic is still possible.

Authenticated Routing for Ad hoc Networks (ARAN) [24] is another secure routing protocol similar to AODV and SAODV. Like SAODV, ARAN assumes asymmetric cryptography key pairs, where signed public keys are available to all nodes. ARAN protects all routing data by signing each packet by every station that forwards it. These signatures then need to be checked by all following nodes. As ARAN route discovery packets (RDPs) do not have a hop count or TTL field, there are no changing parts in these packets and static signatures are sufficient. On the other hand, flooding of RDPs cannot be controlled and always covers the whole network. This imposes a huge overhead on all nodes in the network as these must create and verify large amounts of signatures. Furthermore, lengthening of routes is possible with ARAN.

Ariadne [25] is a secure routing protocol based on DSR. Like SAODV it also uses hashchains for protecting routing messages. Ariadne has three modes of operation, either with pre-established symmetric keys, digital signatures, or using the TESLA system [26] for protecting broadcast messages. Using Ariadne, the destination node of a routing discovery can verify the sender's identity, the originator of a route discovery can authenticate all intermediary nodes and so called per-hop hashing prevents route shortening. All this can be achieved without heavy use of asymmetric cryptography, but it remains unclear how some of the requirements (like synchronized clocks or distributed symmetric keys) should be realized.

Finally, the *Secure Routing Protocol* (SRP) [27], [28], [29], [30] presents a very light-weight solution that again assumes a shared symmetric key between sender and recipient. The overall operation resembles DSR, data is protected by message authentication codes (MACs). In contrast to the extremely low overhead, intermediary nodes are not authenticated, which leaves room for a lot of potential attacks. Furthermore the authors do not explain how keys should be exchanged without established routes.

Table I gives an overview of the different secure MANET routing protocols presented here and compares their properties with our SDSR protocol presented in the next section.

III. SDSR

SDSR is based on the basic DSR functionality as described above. The assumptions made are the presence of only bi-directional links in the network and the existence of so a called MANET-IDs per node. The MANET-ID basically is a signed RSA key-pair that prevents nodes from forging new identities. For details on MANET-IDs, see [19]. Initially, each node only knows its own MANET-ID, distribution of keys to other nodes is integrated in SDSR. In order to handle MANET-IDs and create signatures, nodes must be powerful enough to do asymmetric cryptographics operations in a limited number. This can be assumed even for typical PDAs or smartphones.

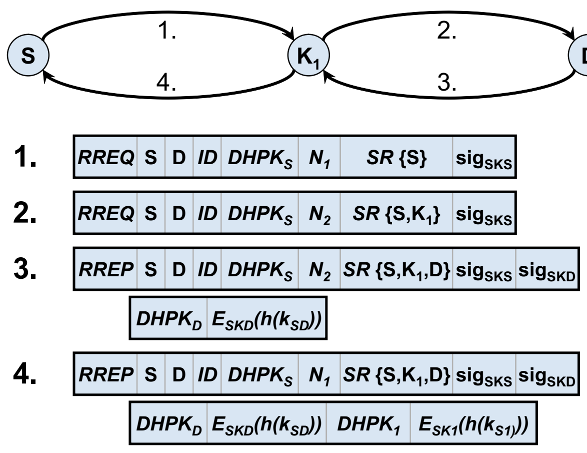
S
K1
D
1.
2.
3.
4.
1. RREQ S D ID DHPK_S N_1 SR {S} sig_SKS
2. RREQ S D ID DHPK_S N_2 SR {S,K1} sig_SKS
3. RREP S D ID DHPK_S N_2 SR {S,K1,D} sig_SKS sig_SKD
DHPK_D E_SKD(h(k_SD))
4. RREP S D ID DHPK_S N_1 SR {S,K1,D} sig_SKS sig_SKD
DHPK_D E_SKD(h(k_SD)) DHPK_1 E_SK1(h(k_S1))

Fig. 1. SDSR Protocol

SDSR adds mechanisms to DSR to

1) secure the integrity of a route
2) secure the freshness of a route
3) secure the authenticity of all nodes participating in a route
4) exchange session keys between source / destination and all other nodes in a route.

We first present a basic version of SDSR and then describe extensions that provide the complete functionality. Figure 1 shows the basic steps in a route discovery where node S searches a route to node D. In step 1, S creates a route request that contains the source S, the node D, a route request ID unique per source, a public Diffie-Hellmann key $DHPK_S$, a random nonce N_1, an initial source route $SR\{S\}$. Finally, S creates a signature sig_{SK_S} using its private MANET-ID key. This signature protects all static information (S, D, ID, and $DHPK_S$) from alteration. D can use this signature to verify, that S actually sent the route request.

Before forwarding a route request (step 2), the intermediary node K_1 appends itself to the source route and transforms the nonce N_1 into N_2 (see below). When the route request finally reaches D, the destination generates a route reply (step 3). This route reply contains all the static information from the route request, the source route, the signature from S and a new signature sig_{SK_D}. This signature includes the source route and sig_{SK_S}. So in the route reply phase (steps 3 & 4), alterations of the source route can be detected. Additionally, all elements from the route request are still protected. By verifying the signature sig_{SK_D}, S can be sure that D created the reply. Finally, D adds its public Diffie-Hellmann key $DHPK_D$ and an encrypted, hashed key k_{SD} to the route reply (see below).

All intermediary nodes, forwarding the route reply towards the destination will undo a part of the nonce transformation (see below) and will also append a public Diffie-Hellmann and an encrypted, hashed key (see below).

There are three problems remaining now:

The source route may be altered during the route request phase. This is prevented by the so-called *nonce transformation.* S chooses a random nonce N_1, each intermediary node K_i chooses a random symmetric key k_i[4], that it keeps secret. It then calculates $N_{i+1} = E_{k_i}(N_i)$, i.e. it encrypts the nonce using its secret key and a symmetric crypto algorithm like AES. During the route reply phase, all these encryptions need to be decrypted in exactly reverse order, i.e. $N_i = D_{k_i}(N_{i+1})$. If S does not receive N_1 in the route reply, the result will be discarded. This ensures that the way during the route request and the way during the route reply must be traversed in exactly the same order. As the way back is mandated by a (signature protected) source route, the way towards D must have been the same.

No session keys have yet been exchanged. The public Diffie-Hellmann keys are used to securely exchange session keys using the Diffie-Hellmann [31] key exchange protocol. The system uses a fixed prime n and $0 < z < n$. $DHPK_S$ is then calculated based on a random number s as $DHPK_S = z^s \bmod n$. All other nodes calculate their $DHPK_x$ accordingly. Now D can calculate a session key k_{SD} as $k_{SD} = DHPK_S^d \bmod n$, S can vice versa calculate $k_{SD} = DHPK_D^s \bmod n$. So S and D have both agreed on a common session key k_{SD} that can e.g. be used to encrypt later data traffic. The same way, S and K_i can establish k_{Si}.

Intermediary nodes may forge identities of other nodes. Up to now, intermediary nodes K_i are not authenticated. S can authenticate the nodes using the signatures in $E_{SK_i}(h(k_{Si}))$. Additionally these fields can be used to verify that the correct session key k_{Si} has been calculated.

Using this protocol, S has reached all of its security goals: it has authenticated D and all intermediary nodes, it is sure that the source route is authentic and corresponds to the way travelled through the network and it has exchanged session keys with D and all intermediary nodes.

[4]e.g. an AES key

IV. Extensions

The former section only presented a simplified version of SDSR. Some extensions are still missing that are only outlined here. For details, again see [19].

E.g. D should also exchange session keys with the intermediary nodes. This can be achieved by doing additional Diffie-Hellmann key exchanges. S need to sent all $DHPK_{K_i}$ to D in an extra packet following step 4, now D can calculate all k_{Di}.

The next extension is called *key distribution* and is responsible for distributing the signed public keys of nodes to other nodes in a route that do not know them yet. Each node has a cache (of limited size) of signed public keys of other nodes. When it needs to verify a signature where the corresponding public key is not known yet, SDSR offers a way for the node to ask the original node for a copy of its signed public key. The route request and reply packets contain request vectors, where the single bits correspond to node positions in the route. If a node needs a certain key, it simply sets the bit of the corresponding node in a route request or route reply packet and the note will append its key to one of the following packets. Again a final packet travelling from node S to D is necessary after step 4.

Another functionality not mentioned yet is route maintenance. When routes break, signed route error messages will be sent from the position of the error towards S and D where a new route request may be initiated. As session keys were already established, an optimized route discovery may be used, like outlined in the following section.

V. Optimizations

MANET routing protocols rely heavily on the use of optimization techniques in order to reduce routing overhead. SDSR does not allow the same optimization techniques that DSR specifies, because some of them would violate security. E.g. the DSR standard [1] describes techniques called "'Caching Overheard Routing Information"' and "'Replying to Route Requests using Cached Routes"' where routing information from packets captured in promiscuous mode is integrated into the own routing database or cached information in this database is used to generate route replies.

All these optimizations cannot be used for SDSR, as they do not provide authentication of nodes or exchange of session keys. There are, however, a number of other optimizations that might be implemented for SDSR. Routing protocol information may e.g. be *piggybacked* on normal data packets, thus reducing the protocol overhead. Data may e.g. be already sent with the route reply packet. Before handing this data to an application, it must be queued by S until all protocol steps have finished and the authenticity of the nodes has been verified.

If intermediary nodes already know a valid route towards S, they may stop further flooding of a packet and use *route request unicasting* to deliver the route request on an unicast route to D. Nonce transformation still has to be done during this unicasting. But this optimization opens another potential attack: the intermediary node may unicast the packet on a suboptimal route, thus allowing the lengthening of routes. [19] describes some ways to prevent this.

Finally, if nodes have already established session keys, they may *reuse secret keys* in further route discoveries, preventing costly asymmetric cryptography in this process.

VI. Validation

When designing cryptographic protocols, care should be taken to verify the correctness of the protocol. One common method to verify whether a given cryptographic protocol actually reaches the desired design goals is the so called *BAN Logic*, named after their designers Burrows, Abadi and Needham [32].

A. Notation

BAN-Logic distinguishes protocol participants (*principals*), *encryption keys* and formulas or *statements*. In the following short notation overview, P, Q, and R denote random participants, K represents a key and X and Y are statements.

$P \mid\equiv X$: **P believes X**
: P behaves, as if X were true.

$P \triangleleft X$: **P sees X**
: P has received message X (from an unspecified sender) and can read X (maybe. after decryption).

$P \mid\sim X$: **P said X**
: This is no statement, whether X has been sent in the current or in an earlier protocol run; when sent, $P \mid\equiv X$ was true.

$P \mid\Rightarrow X$: **P has authority over X**
: P is credible with respect to X; P can e.g. be a server with a special functionality X (key generation, signature etc.).

$\sharp(X)$: **X is fresh**
: X has not been used in any earlier protocol run; X is also called a *nonce*.

1.	$H_1 \rightarrow H_2$	:	$N_1,\ \{ID,\ Y_1\}_{PK_1^{-1}}$
2.	$H_i \rightarrow H_{i+1}$ $\forall\, i = 2 \ldots (n-1)$	:	$N_i = \{N_{i-1}\}_{k_i},\ \{ID,\ Y_1\}_{PK_1^{-1}}$
3.	$H_n \rightarrow H_{n-1}$	:	$\Big\{(H_1, \ldots, H_n),\ \{ID,\ Y_1\}_{PK_1^{-1}}\Big\}_{PK_n^{-1}},\ N_{n-1},\ (Y_n),\ (\{h(k_{(1,n)})\}_{PK_n^{-1}})$
4.	$H_i \rightarrow H_{i-1}$ $\forall\, i = (n-1) \ldots 2$	:	$\Big\{(H_1, \ldots, H_n),\ \{ID,\ Y_1\}_{PK_1^{-1}}\Big\}_{PK_n^{-1}},\ N_{i-1} = \{N_i\}_{k_i^{-1}},\ (Y_n, \ldots, Y_i),\ (\{h(k_{(1,j)})\}_{PK_j^{-1}}\ \forall\, j = n \ldots i)$

TABLE II
BAN FORMALIZATION OF SDSR

$P \overset{K}{\leftrightarrow} Q$: **P and Q share a common secret key K**
Nobody else knows K or can gain access to K, unless P or Q trust him.

$\overset{K}{\mapsto} P$: **K is public key of P**
The corresponding secret key K^{-1} is only known to P.

$P \overset{X}{\rightleftharpoons} Q$: **P and Q share a common secret X**

$\{X\}_K$: **X is encrypted with K**
K may be a symmetric or asymmetric key.

$\langle X \rangle_Y$: **Y proves identity of sender X**
Y may e.g. be a password that is used in a *keyed hash* function to create a signature of X.

B. Deduction Rules

Table III shows the BAN deduction rules. If the conditions on top of a rule are true, the statement below the bar can be derived. For explanations see [32].

As explained in [33], [34], the original BAN rules are not complete. For our following prove we need some additional rules that are intuitively clear:

$$\frac{P \mid\Rightarrow X, \quad \sharp X}{P \mid\equiv \sharp X} \tag{10}$$

If a node P has control over X and if X is fresh (e.g. because it is generated in each protocol run), then P believes in the freshness of X.

$$\frac{P \mid\equiv\ Q \mid\equiv h(X)}{P \mid\equiv\ Q \mid\equiv X} \tag{11}$$

If P believes that Q believes $h(X)$[5], then P believes that Q believes X.

This rules will now be used to prove the correctness of SDSR with respect to the security goals expressed in section III. The first step is a protocol formalization according to the BAN notation.

[5]and h is a cryptographic hashfunction

$$\frac{P \mid\equiv Q \overset{K}{\leftrightarrow} P, \quad P \triangleleft \{X\}_K}{P \mid\equiv (Q \mid\sim X)} \quad \frac{P \mid\equiv \overset{K}{\mapsto} Q, \quad P \triangleleft \{X\}_{K^{-1}}}{P \mid\equiv (Q \mid\sim X)} \tag{1}$$

$$\frac{P \mid\equiv \sharp(X), \quad P \mid\equiv (Q \mid\sim X)}{P \mid\equiv (Q \mid\equiv X)} \tag{2}$$

$$\frac{P \mid\equiv (Q \mid\Rightarrow X), \quad P \mid\equiv (Q \mid\equiv X)}{P \mid\equiv X} \tag{3}$$

$$\frac{P \mid\equiv X, \quad P \mid\equiv Y}{P \mid\equiv (X, Y)} \quad \frac{P \mid\equiv (X, Y)}{P \mid\equiv X} \quad \frac{P \mid\equiv (Q \mid\equiv (X, Y))}{P \mid\equiv (Q \mid\equiv X)} \tag{4}$$

$$\frac{P \mid\equiv (Q \mid\sim (X, Y))}{P \mid\equiv (Q \mid\sim X)} \quad \frac{P \triangleleft (X, Y)}{P \triangleleft X} \quad \frac{P \mid\equiv \sharp(X)}{P \mid\equiv \sharp(X, Y)} \tag{5}$$

$$\frac{P \triangleleft \langle X \rangle_Y}{P \triangleleft X} \quad \frac{P \mid\equiv (Q \overset{K}{\leftrightarrow} P), \quad P \triangleleft \{X\}_K}{P \triangleleft X} \tag{6}$$

$$\frac{P \mid\equiv (\overset{K}{\mapsto} P), \quad P \triangleleft \{X\}_K}{P \triangleleft X} \quad \frac{P \mid\equiv (\overset{K}{\mapsto} Q), \quad P \triangleleft \{X\}_{K^{-1}}}{P \triangleleft X} \tag{7}$$

$$\frac{P \mid\equiv (R \overset{K}{\leftrightarrow} R')}{P \mid\equiv (R' \overset{K}{\leftrightarrow} R)} \quad \frac{P \mid\equiv (Q \mid\equiv (R \overset{K}{\leftrightarrow} R'))}{P \mid\equiv (Q \mid\equiv R' \overset{K}{\leftrightarrow} R))} \tag{8}$$

$$\frac{P \mid\equiv (R \overset{X}{\rightleftharpoons} R')}{P \mid\equiv (R' \overset{X}{\rightleftharpoons} R)} \quad \frac{P \mid\equiv (Q \mid\equiv (R \overset{X}{\rightleftharpoons} R'))}{P \mid\equiv (Q \mid\equiv (R' \overset{X}{\rightleftharpoons} R))} \tag{9}$$

TABLE III
BAN DEDUCTION RULES

C. Formalization of SDSR

Let $H_1, \ldots, H_n$ be the nodes forming a route from H_1 to H_n. N_1 is a random nonce generated by H_1, the other $N_i \,|\, i = 2 \ldots n$ are calculated using an encryption function like explained earlier. *ID* is a unique Route-Discovery ID. Y_i denotes the public Diffie-Hellmann key of H_i. PK_i is the public key of H_i, PK_i^{-1} the corresponding private key. In addition, each node H_i owns a secret key k_i that is not known to anyone else. k_i^{-1} is used to decrypt messages encrypted by k_i. The DSR protocol can be expressed as shown in table II.

Step 1 describes the transmission of a Route Re-

$H_1 \mid\Rightarrow N_1$	:	H_1 creates the nonce N_1 randomly and newly for each protocol run.
$H_1 \mid\Rightarrow ID$	:	H_1 creates the route request ID newly for each protocol run.
$\overset{Y_i}{\mapsto} H_i$ $\forall\, i = 1 \ldots n$	:	Each node H_i owns a key pair for the DH-protocol (secret: y_i, public: Y_i)
$H_i \mid\equiv \sharp(y_i, Y_i)$ $\forall\, i = 1 \ldots n$	:	Each node H_i creates the DH-keypair newly and randomly for each protocol run.
$H_i \mid\equiv H_i \overset{k_i}{\leftrightarrow} H_i$ $\forall\, i = 2 \ldots (n-1)$	:	Each intermediary node owns a secret key k_i, known only to him.
$\overset{PK_i}{\mapsto} H_i$ $\forall\, i = 1 \ldots n$	:	Each node owns a MANET-ID key pair.
$H_i \mid\equiv \overset{PK_j}{\mapsto} H_j$ $\forall\, i, j = 1 \ldots n$	:	Each node knows the public keys of all other nodes and can verify them.
According to rule 10 this leads to:		
$H_1 \mid\equiv \sharp N_1$	:	As H_1 creates the nonce, it is also convinced of its freshness.
$H_1 \mid\equiv \sharp ID$	:	As H_1 creates ID, it is also convinced of its freshness.

TABLE IV
Initial Conditions

Step 1: $H_2 \triangleleft N_1,\ \{ID,\, Y_1\}_{PK_1^{-1}} \overset{(1)}{\Rightarrow} H_2 \mid\equiv H_1 \mid\sim ID, Y_1$

Step 2: $H_{i+1} \triangleleft N_i,\ \{ID,\, Y_1\}_{PK_1^{-1}} \overset{(1)}{\Rightarrow} H_{i+1} \mid\equiv H_1 \mid\sim ID, Y_1$
$\forall i = 2 \ldots (n-1)$

Step 3: $H_{n-1} \triangleleft \left\{(H_1, \ldots, H_n),\ \{ID,\, Y_1\}_{PK_1^{-1}}\right\}_{PK_n^{-1}},$
$N_{n-1},\ (Y_n),\ (\{h(k_{(1,n)})\}_{PK_n^{-1}})$

$\overset{(1)}{\Rightarrow}$ $H_{n-1} \mid\equiv H_n \mid\sim (H_1, \ldots, H_n),\ \{ID,\, Y_1\}_{PK_1^{-1}}$ and

$\overset{(1)}{\Rightarrow}$ $H_{n-1} \mid\equiv H_1 \mid\sim ID,\, Y_1$ and

$\overset{(1)}{\Rightarrow}$ $H_{n-1} \mid\equiv H_n \mid\sim h(k_{(1,n)})$

Step 4: $H_{i-1} \triangleleft \left\{(H_1, \ldots, H_n),\ \{ID,\, Y_1\}_{PK_1^{-1}}\right\}_{PK_n^{-1}},\ N_{i-1},\ (Y_n, \ldots, Y_i),$
$(\{h(k_{(1,j)})\}_{PK_j^{-1}}\ \forall j = n \ldots i) \quad \forall i = (n-1) \ldots 2$

$\overset{(1)}{\Rightarrow}$ $H_{i-1} \mid\equiv H_n \mid\sim (H_1, \ldots, H_n),\ \{ID,\, Y_1\}_{PK_1^{-1}}$ and

$\overset{(1)}{\Rightarrow}$ $H_{i-1} \mid\equiv H_1 \mid\sim ID,\, Y_1$ and

$\overset{(1)}{\Rightarrow}$ $H_{i-1} \mid\equiv H_j \mid\sim h(k_{(1,j)})\ \forall j = n \ldots i$

TABLE V
Postconditions

$$H_1 \mid\equiv \sharp(ID) \ \wedge\ H_1 \mid\equiv H_n \mid\sim (H_1, \ldots, H_n),\ \{ID,\, Y_1\}_{PK_1^{-1}} \overset{(5)(2)}{\Longrightarrow} H_1 \mid\equiv H_n \mid\equiv (H_1, \ldots, H_n) \tag{12}$$

$$H_1 \triangleleft N_1 \ \wedge\ H_1 \mid\equiv \sharp(N_1) \ \wedge\ H_i \mid\equiv H_i \overset{k_i}{\leftrightarrow} H_i \Rightarrow \left\{\left\{\left\{\left\{\left\{\{N_1\}_{k_2}\right\}_{k_3} \cdots\right\}_{k_{n-1}}\right\}_{k_{n-1}^{-1}} \cdots\right\}_{k_3^{-1}}\right\}_{k_2^{-1}} \tag{13}$$

$$\forall\, i = 2 \ldots (n-1)$$

$\mathbf{H_1}$:

$$H_1 \mid\equiv \sharp(y_1) \ \wedge\ H_1 \mid\equiv H_i \mid\sim h(k_{(1,i)})$$

$$\overset{DH,(11)}{\Rightarrow} H_1 \mid\equiv \sharp(k_{(1,i)})$$

$\mathbf{H_i}$:

$$H_i \mid\equiv \sharp(y_i) \ \wedge\ H_i \mid\equiv H_1 \mid\sim Y_1$$

$$\overset{DH}{\Rightarrow} H_i \mid\equiv \sharp(k_{(1,i)})$$

$$\Rightarrow H_1 \mid\equiv H_1 \overset{k_{(1,i)}}{\leftrightarrow} H_i \ \wedge\ H_i \mid\equiv H_1 \overset{k_{(1,i)}}{\leftrightarrow} H_i$$

TABLE VI
Deductions

quest from node H_1 to node H_2.

Step 2 describes the following transmissions from H_2 to H_3, H_3 to H_4 etc. until H_n where the nonce N_{i-1} is transformed to N_i.

Step 3 describes the transmission of a Route Reply from H_n to H_{n-1}. H_n secures the Source-Route $(H_1, \ldots, H_n)$, ID, and Y_1 by its signature. In addition, the message contains the public Diffie-Hellmann key Y_n and a hashed version of the session key $k_{(1,n)}$, signed by H_n.

In step 4 the Route Reply is transfered back to node 1 following the source route. Nonce-transformations are reversed and public Diffie-Hellmann keys Y_i and signed, hashed keys $k_{(1,i)}$ are appended.

This concludes protocol formalization. Next we need to define initial conditions.

D. Initial Conditions and Postconditions

Table IV shows the initial assumptions which are believed to be true. Each protocol step implies the conclusions shown in table V.

E. Apply Deduction Rules

Given the initial and postconditions one can deduce additional conclusions.

According to equation 12 in table VI after a protocol run, H_1 is sure, that H_n has replied to his current route request with the source route $(H_1, \ldots, H_n)$. But there still is the risk that an intermediary node has manipulated the source route in the route request phase.

Equation 13 shows that H_1 can also exclude this. When H_1 receives N_1 again at the end of the route discovery, this shows that encryptions and decryptions with the symmetric keys k_i were done in exactly the same order. So as long as each node keeps its key secret, this means that all nodes from the way towards H_n have been traversed on the way back in exactly reverse order. But this order is determined by the source route in the route reply.

Now we still cannot tell anything about the identities of intermediary nodes, because no authentication has happened yet. H_1 does this by checking whether the session keys created by the Diffie-Hellmann key exchange correspond to the signed hash values transmitted in the route reply. Let y_i be the secret key of the Diffie-Hellmann key exchange (corresponding to the public key Y_i).

See the last part of table VI for the final deduction. As H_1 contributes a fresh part y_1 to these keys, it can be certain that the resulting key will also be fresh in case of a correct Diffie-Hellmann key exchange. As soon as H_1 has calculated the common key $k_{(1,i)}$, it can use the signed hash value to check, if its partner in the Diffie-Hellmann key exchange has actually been H_i and if this node has calculated the same key than itself.

As H_i also contributes a fresh part y_i to the key, he can also be sure of the freshness of the key $k_{(1,i)}$. It can also be sure to have received a valid Y_1 from node H_1. Y_1 is not proven to be fresh, but in case of a replay, H_1 and H_i will calculate different $k_{(1,i)}$. H_1 will detect this when verifying the key and will discard it. Then a new route discovery becomes necessary.

So we can finally state that all security goals as expressed above are actually met by SDSR. Now H_1 can use the source route and the exchanged session keys to communicate with the other nodes.

VII. Summary and Outlook

As we have seen, SDSR is a powerful and secure routing protocol for MANETs that provides more functionality than other approaches. In a following publication we will publish simulation and analytical results showing that the resource consumption of SDSR is nevertheless modest and in the dimension of the other protocols. Furthermore we will discuss additional aspects of integration SDSR in the overall architecture SAM.

References

[1] D. B. Johnson, D. A. Maltz, Y.-C. Hu, and J. G. Jetcheva, "The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR)," http://www.ietf.org/internet-drafts/draft-ietf-manet-dsr-09.txt, Apr. 2003.

[2] F. Kargl, A. Klenk, S. Schlott, and M. Weber, "Sensors for Detection of Misbehaving Nodes in MANETs," in *Proceedings of Workshop Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA 2004)*, Dortmund, Germany, July 2004.

[3] ——, "Advanced Detection of Selfish or Malicious Nodes in Ad hoc Networks," in *Proceedings of 1st European Workshop on Security in Ad-Hoc and Sensor Networks (ESAS 2004)*, ser. Springer LNCS, Heidelberg, Germany, Aug. 2004.

[4] L. Zhou and Z. J. Haas, "Securing Ad Hoc Networks," *IEEE Network*, vol. 13, no. 6, pp. 24–30, 1999, also available as http://citeseer.nj.nec.com/zhou99securing.html.

[5] J.-P. Hubaux, L. Buttyán, and S. Čapkun, "The Quest for Security in Mobile Ad Hoc Networks," in *Proceeding of the ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHOC)*, 2001, also available as http://citeseer.nj.nec.com/493788.html.

[6] L. Buttyán and J.-P. Hubaux, "Nuglets: a Virtual Currency to Stimulate Cooperation in Self-Organized Mobile Ad Hoc Networks," EPFL-DI-ICA, Tech. Rep. DSC/2001/001, Jan. 2001.

[7] ——, "Stimulating Cooperation in Self-Organizing Mobile Ad Hoc Networks," *ACM/Kluwer Mobile Networks and Applications*, vol. 8, no. 5, Oct. 2003.

[8] R. Anderson and M. Kuhn, "Tamper Resistance - a Cautionary Note," in *Proceedings of the Second Usenix Workshop on Electronic Commerce*, Nov. 1996, pp. 1–11, also available as http://citeseer.nj.nec.com/article/anderson96tamper.html.

[9] ——, "Low cost attacks on tamper resistant devices," in *IWSP: International Workshop on Security Protocols, LNCS*, 1997, also available as http://citeseer.nj.nec.com/anderson97low.html.

[10] S. Zhong, J. Chen, and Y. R. Yang, "Sprite: A simple, cheat-proof, credit-based system for mobile ad-hoc networks," in *Proceedings of IEEE Infocom '03*, San Francisco, CA, Apr. 2003.

[11] S. Marti, T. J. Giuli, K. Lai, and M. Baker, "Mitigating routing misbehavior in mobile ad hoc networks," in *Mobile Computing and Networking*, 2000, pp. 255–265, also available as http://citeseer.nj.nec.com/marti00mitigating.html.

[12] Y. Zhang and W. Lee, "Intrusion detection in wireless ad-hoc networks," in *Mobile Computing and Networking*, 2000, pp. 275–283, also available as http://citeseer.nj.nec.com/zhang00intrusion.html.

[13] Y. Zhang, W. Lee, and Y.-A. Huang, "Intrusion Detection Techniques for Mobile Wireless Networks," *to appear in ACM Wireless Networks (WINET)*, vol. 9, 2003, also available as http://www.wins.hrl.com/people/ygz/papers/winet03.pdf.

[14] P. Michiardi and R. Molva, "Prevention of Denial of Service attacks and Selfishness in Mobile Ad Hoc Networks," http://www.eurecom.fr/ michiard/pub/michiardi_adhoc_dos.ps.

[15] ——, "A Collaborative Reputation mechanism to enforce node cooperation in Mobile Ad Hoc Networks," in *Proceedings of the 6th IFIP Communication and Multimedia Security Conference*, Portorosz, Slovenia, Sept. 2002.

[16] S. Buchegger and J.-Y. L. Boudec, "Nodes Bearing Grudges: Towards Routing Security, Fairness, and Robustness in Mobile Ad Hoc Networks," in *Proceedings of the Tenth Euromicro Workshop on Parallel, Distributed and Network-based Processing*. Canary Islands, Spain: IEEE Computer Society, Jan. 2002, pp. 403–410, http://citeseer.nj.nec.com/article/buchegger02nodes.html.

[17] ——, "Performance Analysis of the CONFIDANT Protocol: Cooperation Of Nodes - Fairness in Distributed Adhoc Networks," in *Proceedings of IEEE/ACM Workshop on Mobile Ad Hoc Networking and Computing (MobiHOC)*, Lausanne, CH, June 2002.

[18] F. Kargl, S. Schlott, M. Weber, A. Klenk, and A. Geiß, "Securing Ad hoc Routing Protocols," in *Proceedings of 30th Euromicro Conference*, Rennes, France, Aug. 2004.

[19] F. Kargl, "Sicherheit in Mobilen Ad hoc Netzwerken," Ph.D. dissertation, University of Ulm, Ulm, Germany, 2003, also available as http://medien.informatik.uni-ulm.de/~frank/research/dissertation.pdf.

[20] M. Guerrero Zapata, "Secure Ad hoc On-Demand Distance Vector Routing," *ACM Mobile Computing and Communications Review (MC2R)*, vol. 6, no. 3, pp. 106–107, July 2002, also available as http://doi.acm.org/10.1145/581291.581312.

[21] M. Guerrero Zapata and N. Asokan, "Securing Ad hoc Routing Protocols," in *Proceedings of the 2002 ACM Workshop on Wireless Security (WiSe 2002)*, Sept. 2002, pp. 1–10, also available as http://doi.acm.org/10.1145/570681.570682.

[22] M. Guerrero Zapata, "Secure Ad hoc On-Demand Distance Vector (SAODV) Routing," http://ant.eupvg.upc.es/~tarom/draft-guerrero-manet-saodv-00.txt, Aug. 2002.

[23] C. E. Perkins, E. M. Royer, and S. Das, "RFC 3561: Ad Hoc On Demand Distance Vector (AODV) Routing," http://www.ietf.org/rfc/rfc3561, July 2003.

[24] 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 2002 IEEE International Conference on Network Protocols (ICNP)*, Nov. 2002, also available as http://signl.cs.umass.edu/pubs/aran.icnp02.ps.

[25] Y.-C. Hu, A. Perrig, and D. B. Johnson, "Ariadne: A secure On-Demand Routing Protocol for Ad hoc Networks," in *Proceedings of MobiCom 2002*, Atlanta, Georgia, USA, Sept. 2002.

[26] A. Perrig, R. Canetti, J. Tygar, and D. Song, "The TESLA Broadcast Authentication Protocol," *RSA CryptoBytes*, vol. 5 (Summer), 2002.

[27] P. Papadimitratos and Z. J. Haas, "Secure Routing for Mobile Ad hoc Networks," in *SCS Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS 2002)*, San Antonio, TX, Jan. 2002, also available as http://wnl.ece.cornell.edu/Publications/cnds02.pdf.

[28] P. Papadimitratos, Z. J. Haas, and P. Samar, "The Secure Routing Protocol (SRP) for Ad Hoc Networks," draft-papadimitratos-secure-routing-protocol-00.txt, Dec. 2002.

[29] P. Papadimitratos and Z. Haas, "Performance Evaluation of Secure Routing Protocol for Mobile Ad Hoc Networks," in *First ACM Workshop on Wireless Security (WiSe), Poster Session, in conjunction with ACM MobiCom 2002*, 2002, also available as http://people.cornell.edu/pages/pp59/Docs/wise02.pdf.

[30] P. Papadimitratos and Z. J. Haas, "Secure Link State Routing for Mobile Ad Hoc Networks," in *IEEE Workshop on Security and Assurance in Ad hoc Networks, in conjunction with the 2003 International Symposium on Applications and the Internet*, Orlando, FL, Jan. 2003.

[31] W. Diffie and M. Hellmann, "New Directions in Cryptography," *IEEE Transactions on Information Theory IT*, vol. 22, no. 6, pp. 644–654, 1976.

[32] M. Burrows, M. Abadi, and R. M. Needham, "A Logic of Authentication," *ACM Transactions on Computer Systems*, vol. 8, no. 1, pp. 18–36, 1990.

[33] D. M. Nessett, "A Critique of the Burrows, Abadi and Needham Logic," *ACM Operating Systems Review*, vol. 24, no. 2, pp. 35–38, 1990.

[34] M. Burrows, M. Abadi, and R. M. Needham, "Rejoinder to nessett," *ACM Operating Systems Review*, vol. 24, no. 2, pp. 39–40, 1990.