SEAMAN: A Security-Enabled Anonymous MANET Protocol

Harald H.-J. Bongartz, Tobias Ginzler, Thomas Bachran
Research Institute for Communication,
Information Processing, and Ergonomics (FKIE)
Research Establishment for Applied Sciences (FGAN)
Neuenahrer Straße 20, D-53343 Wachtberg
{bongartz, ginzler, bachran}@fgan.de

Pere Tuset
Escola Universitària Politècnica de Mataró (EUPMt)
Universitat Politècnica de Catalunya
Av. Puig i Cadafalch 101-111, E-08303 Mataró
tuspeipe@eupmt.upc.edu

ABSTRACT

Mobile ad hoc networks or MANETs are supposed to play an important role in future military communication. MANETs offer quickly deployable network functionality even when the existing infrastructure is damaged or unusable. But while offering many advantages, the MANET technologies today are facing new security threats. These threats arise from the simplicity to eavesdrop on the wireless medium and launch direct attacks against the routing protocol. A MANET protocol concept that deals with these challenges is presented in this paper.

1.0 INTRODUCTION

Mobile wireless ad hoc networks (MANETs) have interesting characteristics, both for the civilian and the military domains. MANETs automatically adapt to the environment and provide reliable communication. They are self-organising, failure resistant and are able to react on frequent topology changes caused by node mobility, varying radio conditions or hostile intervention. But the development of MANETs often neglects two requirements which are crucial for tactical networks: security and multicast. During the first stage of network and routing protocol design, the focus is mostly on functionality and performance. Security aspects are usually left for later consideration. This often leads to incomplete or inefficient security solutions.

In some cases, MANET security is delegated as a task for upper layer protocols. But upper layers cannot protect the routing information and, with the possibility to forge routing messages, an adversary can easily re-route traffic and drop frames. For this reason it is crucial to have authenticated routing traffic. Also, the information contained in the management traffic can be used in a passive attack to reconstruct the network topology and analyse movement and traffic patterns. This is unacceptable in tactical networks, therefore it is important to hide node identities and traffic flow information from adversaries. Sometimes, simple mechanisms like pre-shared keys are used to establish a certain level of security, but this approach suffers from the threat of key compromise.

Multicasting is a communication scheme to efficiently send data from one sender to multiple recipients. It is widely used in the military domain and the NATO NEC Feasibility Study [1] states that its support is a design requirement for tactical MANETs. An efficient solution can reduce network load significantly so that it is particularly valuable for wireless networks with limited available bandwidth.

In this paper we propose a mechanism that combines

- an efficient multicast multi-hop ad hoc routing protocol,
- a detection mechanism for single foreign hosts and foreign networks that want to join a secured MANET (*Foreigner Detection*),
- an anonymous clear-text protocol to authenticate foreign hosts and networks (*Anonymous Authentication*)
- a frame encryption scheme that uses a common key to encrypt all MANET traffic, and
- a key management system to dynamically generate and update the common key.

We introduce this concept under the name of SEAMAN, **S**ecurity-**E**nabled **A**nonymous **MAN**ET protocol. SEAMAN provides a high level of security while benefiting from the advantages of MANETs. The concept itself does not rely on specific routing or key management protocols. Instead, it describes how these instances can interact to overcome the existing limitations. Our approach addresses authentication, confidentiality and anonymity of mobile ad hoc networks. Efficient secure multicast traffic is supported.

2.0 RELATED WORK

Nowadays several different approaches aim to enhance the security aspects of mobile ad hoc networks. Each approach focuses on a security attribute or a group of attributes that are considered to be the most relevant ones, e. g. authentication, authorization and confidentiality, depending on the design requirements.

There are some papers that discuss security-enabled MANET multicast protocols or principles for multicast security in MANETs. A very in-depth discussion of possible attacks on MAODV (Multicast-extended AODV) routing can be found in [2]. The authors also propose an authentication framework to protect an MAODV network against these attacks, but the framework is quite specific to MAODV and does not consider confidentiality of routing and data messages. [3] introduces a protocol for secure communication in multicast groups which is built on top of existing unicast MANET routing protocols. This approach uses a pair of multicast trees for each multicast group; one for security information and the other for the group's data traffic. It takes the authentication process, key establishment and encryption process into account but, since it does not cover security aspects for the underlying MANET protocol, the solution is not sufficient for our purposes. Some of the proposed concepts might be applicable for internal confidentiality and might be regarded in upper layer protocols. In [4], Galera et al. extend their multicast MANET protocol MMARP with digital signatures using a public key scheme. To avoid the need for a Certification Agency (CA), they include the public keys of the originator and the sender in every message. To keep attackers from impersonating other nodes, they use Cryptographically Generated Addresses (CGA), but a problem remains that, in the absence of certificates, either the CGAs or the corresponding public keys still must be authenticated as trusted network members by an external source. The approach does not include frame encryption or protection against reconnaissance or replay attacks. The BSMR protocol proposed in [5] shows how to secure the multicast tree of a tree-based multicast protocol against insider attacks, but the authors do not discuss external attacks.

There are more sources for security in MANETs that deal with unicast traffic only. To deal with route anonymity and location privacy, the authors of ANODR [6] propose to use the broadcast with a trapdoor information concept. ASR [7] describes how to ensure the identity and location privacy of mobile nodes and secure the discovery of routes. The authors of the MASK paper [8] describe how to achieve node unlocability and untrackability, as well as end-to-end flow untraceability, to provide anonymous communications in Mobile Ad hoc Networks. A more detailed description of MANET security attributes and an extended summary of the reviewed security-enabled protocols can be found in [9].

To our knowledge none of the existing protocols simultaneously addresses all the security requirements for a tactical MANET to a proper extent. It is our goal to fill this gap with a protocol that provides node authentication, data integrity and external confidentiality, reduced reconnaissance, node and flow path untraceability for efficient multicast transfers in mobile wireless ad hoc networks.

3.0 PROBLEM STATEMENT

We consider an application of mobile ad hoc networks in scenarios with a high demand of confidentiality, message and network integrity and resilience against external attacks. Tactical military operations, police and emergency response operations demand such networks. One interesting application is the operation of multi-robot systems in military and civilian emergency scenarios. Immediate and secure control of robot motion and attained sensor data can be of crucial importance for operational success.

3.1 Networking aspects

We assume that no more than about 50 participants (a platoon or similarly sized civilian unit) will have a demand for direct communication with each other in a MANET. Energy-constraints have been taken into account in the design process but are not of crucial importance for our considerations. It is presumed that the communication equipment has a power supply that allows for continuous operation during deployment. This is in contrast to a wireless sensor network, where energy constraints are of utmost importance and radio operation has to be restricted. Energy consumption for cryptographic calculations is considered to be of minor importance when compared to the energy consumed by the radio. We nevertheless take into account that an attacker might want to provoke a power drain by deliberately consuming network and node resources.

During operation, single participants and groups of participants may join or leave the area of interest. Also, personnel might get separated from the unit or communication equipment might get lost and compromised. To ensure secrecy, this leads to a number of network operations that must be supported:

Join operation:
A single node joins a secured network. Before the join operation is performed, the node should not be able to communicate with the secured network.

Leave operation:
A single node leaves a secured network. After leaving, the node should not be able to communicate with the other nodes.

Eject operation:
A single or multiple nodes are excluded from the secured network, e. g. due to a presumed hostile take-over. After ejection, the node(s) must not be able to communicate with other nodes in the network.

Merge operation:
Two secured networks merge to form a new common secured network. The separate networks should be able to communicate within themselves, but not with each other before the merge. The *Join* operation can be considered a special case of a *Merge*, where one network only consists of a single node.

Split operation:
The network is split up, e. g. when a group of participants moves out of the area. After the split, the separate networks should still be able to communicate within themselves, but not with members of the other network. The *Leave* operation can be considered a special case of a *Split*, where one of the networks only consists of a single node.

3.2 Security aspects

From a security point of view, it is useful to distinguish between authorised, *internal* MANET nodes and unauthorised, *external* nodes. Internal security protects internal nodes against other internal nodes, while external security protects internal nodes from external ones. External attacks, especially in military scenarios, are passive reconnaissance attempts with the goal to disclose node identities or traffic flows. Identification of conspicuous sources or sinks of information is valuable for an attacker. The protection against such threats is called external anonymity. Internal confidentiality, authenticity and integrity is often achieved with pair-wise session keys for every radio link. But the establishment of session keys is expensive. It also conflicts with the use of effective multicast mechanisms that rely on the broadcast property of the wireless medium. Another possibility to establish internal security is to use Public Key algorithms in combination with node certificates. This imposes high processing load to the nodes.

From our point of view it is most important for a MANET protocol to guarantee *external* anonymity, confidentiality, authenticity and integrity of all user generated traffic and routing messages. Many aspects of *internal* security are better handled in upper layer protocols like IP Security. Nevertheless, it is necessary to have powerful mechanisms to react on internal threats. In particular it should be possible to exclude detected internal attackers permanently. For this reason the concept should be combined with an Intrusion Detection System to identify and report internal security threats and a revocation mechanism to withdraw access permissions.

A key management system is necessary to administer the encryption keys in a MANET. The key management system is responsible to ensure that only authorised users get the MANET key (*key secrecy*). The system also has to ensure that it is impossible to decrypt messages which were send before a node joined the MANET (*backward secrecy*). Additionally, it is responsible to exclude leaving members from future communication (*forward secrecy*). An important property of the key management system is the possibility to eject a node or group of nodes on demand, e. g. by triggering a rekeying operation. Such an ejection is necessary if a node is captured by an adversary.

The simplest form of key management system is to manually set the key at every MANET node. It is obvious that this concept is only suitable for static and small groups, as it doesn't scale well. Both assumptions are not fulfilled in a MANET. Most key management systems today rely on a dedicated server, but this approach cannot be used in tactical environments. For tactical MANETs robustness and resilience against node failure is crucial. Single points of failure have to be avoided by all means. Additionally, the key management should pay attention to the restricted resources in a MANET and use them efficiently. Therefore, the key management system should be decentralised, support multicast and be robust against a large variety of attacks.

In short, a secure MANET should be resistant against replay attacks, cryptanalysis, attacks against the routing protocol, and denial of service attacks (DoS). It is desirable to protect the nodes' identities and their movement pattern against reconnaissance. Additionally, the data flow paths should be hidden from adversaries to prevent identification of critical nodes.

4.0 PROTOCOL CONCEPT

In this section we propose SEAMAN, a protocol that ensures anonymity and secrecy of all messages against external adversaries. To achieve this goal several techniques are applied. These techniques are explained in detail in the following subsections.

4.1 Encryption on Link Layer

Most MANET protocols can only take care of their own management frames to protect the routing itself from external influence, e. g. by signing and/or encrypting routing messages. They have no influence on the format and visibility of frames that are sent over the wireless interface. SEAMAN in contrast uses MAC layer forwarding. We use the terms routing and forwarding synonymously in this paper. SEAMAN handles every frame sent over the wireless interface, not only its own management frames and also controls the information contained in the link layer headers. It uses a dynamic MANET key to encrypt every frame before delivering it to the wireless interface. In addition, a hashed message authentication code (HMAC) is appended to the frame. For every transmission in a multi-hop route, the frame content and thus the encrypted frame changes due to a sequence number contained in the frame header. So, an external attacker cannot identify a forwarded frame.

Optionally, every frame can be padded with arbitrary data to conceal the original length. Additionally, random delays between retransmissions and dummy traffic can be introduced to make the reconnaissance of multi-hop paths even harder. This renders it difficult for an attacker to trace a frame in a multi-hop path.

While frame encryption increases the computational overhead for the message transmission, the latter options reduce the actual available bandwidth. Application of these techniques is considered a security trade-off.

4.2 Anonymity on Link Layer

The MANET forwarding protocol used in SEAMAN sends all data frames in broadcast mode, concealing sender and receiver addresses of the frame. This is done by setting both addresses to the broadcast address or another reserved address. In fact, the SEAMAN protocol only needs one flag in clear text visible in a frame to distinguish between MANET key encrypted frames and frames encrypted with a temporary key that is used during a network merge operation. In general, using a clear text link layer header eases detection of SEAMAN frames by nodes and thus increases protocol efficiency. Aside from this, depending on the radio technology used, additional link layer headers can be avoided completely.

In combination with the link layer encryption, a passive attacker has no direct information about sources or destinations of data streams. Adding data padding, random retransmission delays and dummy traffic decreases the applicability of traffic flow analysis dramatically with the drawbacks mentioned above.

4.3 Foreigner Detection

Since all traffic in SEAMAN is encrypted on the link layer with a dynamic key, a node that does not know this MANET key cannot be easily integrated into the MANET. The same situations applies for two networks using different MANET keys that want to merge. Therefore it is required to detect an approaching node and contact it for authentication. For this, we introduce the following *Foreigner Detection* mechanism.

1. If a node A, being a member of MANET $\mathcal{A}$ receives a message frame from another node B, it first checks if it is SEAMAN frame. This is done by comparing the sender link layer address with the common address used in SEAMAN. This check prevents a reaction on messages sent by foreign network nodes that do not speak the SEAMAN protocol. In addition, A checks that the *Bridge Flag* in the frame is not set. The purpose of the Bridge Flag will be described later.

2. If the message frame is a SEAMAN frame, node A tries to decrypt it by using the MANET key valid in $\mathcal{A}$. There may be more than one MANET key valid at any moment due to key updates, as will be discused later.

3. If decryption succeeds, node B belongs to network $\mathcal{A}$, and the frame content is analysed and handled as defined by the MANET forwarding protocol. Replay attacks by foreign nodes should be handled by the MANET forwarding protocol, e. g. by detecting repeated sequence numbers in the decrypted frames.

4. If decryption fails, the protocol assumes that B is either

   (a) a node belonging to another SEAMAN network $\mathcal{B}$, or

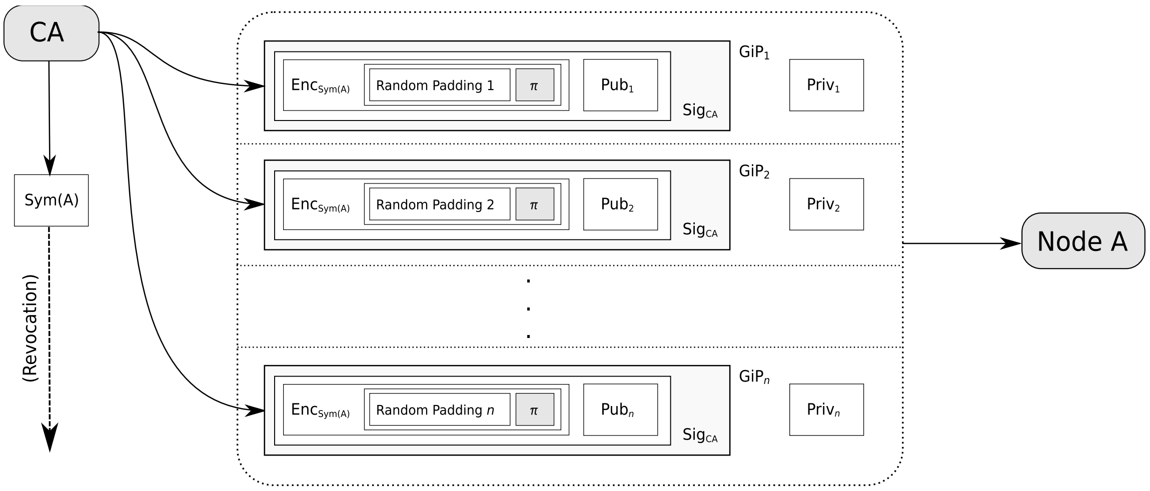

Figure 1: A CA generates a list of Group-identifiable Pseudonyms (GiP) for Node A

(b) a solitary SEAMAN node.

A now tries to authenticate B as described in the following section.

4.4 Anonymous Authentication

Messages sent by a node must not contain any information that allows the identification of the sender or the sender's group by unauthenticated nodes. This security policy also applies for unencrypted messages used to authenticate other nodes.

To fulfil this requirement, we chain two anonymous authentication methods: a perfectly anonymous authentication using a secret handshake protocol; and an authentication method based on public key cryptography that reveals the identity of a certification authority, but not the node identity itself.

Perfect Pseudonyms: The secret handshake from the MASK protocol [8] uses a Trusted Authority (TA) to generate lists of pseudonyms for every node using pairing-based cryptography. The pseudonyms do neither reveal any information about the node itself, nor about the TA that issued them, hence we call them *Perfect Pseudonyms (PP)*. The PPs can be used to prove the membership in a group defined by the TA to other nodes that use PPs from the same TA, e. g. based on the same tuple of generating secrets. A node can have an arbitrarily large number of PPs and should use every pseudonym only once for authentication. But this poses a new problem: Even though the TA could revoke every single PP for a compromised node, the number of PPs makes a check against a revocation list impractical.

Group-identifiable Pseudonyms: In addition to the Perfect Pseudonyms, we introduce *Group-identifiable Pseudonyms (GiP)*. A Certification Authority (CA) generates lists of public-private key pairs and a single, secret symmetric key for every node. Then, for every key pair, a random padding and a publicly known value (e.g. a fixed number of digits from π) are combined and encrypted with that symmetric key, resulting in a pseudo-random 'blob'. The public key is then appended to that blob and the result is signed with the secret key of the CA. Now we have a list of CA-signed pseudonyms, each consisting of a blob and a single public key (figure 1). The signature makes these pseudonyms identifiable with respect to the CA. But the symmetric key is the only entity which can associate the pseudonyms with a certain node and it is kept within the CA. When the GiPs for a node should be revoked, the CA publishes the symmetric key that was used for the encryption of the blob in every single GiP. Now, every node can use that symmetric key to check received GiPs for validity. Usually, any certificate associated with that node would also be revoked by that CA. SEAMAN itself does not handle the distribution of revocation lists.

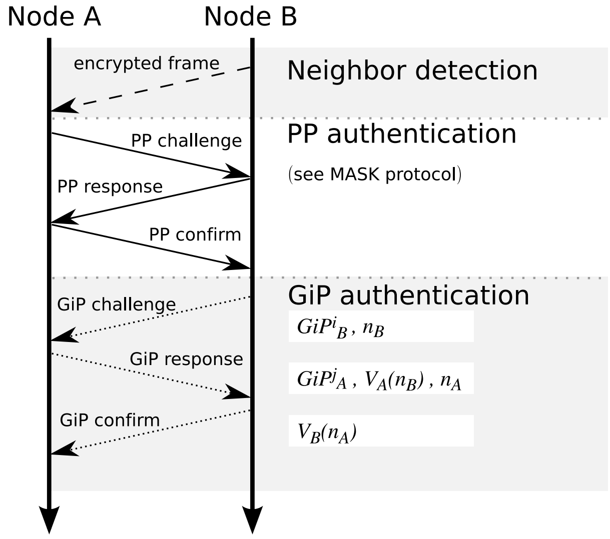

Figure 2: The authentication sequence between nodes A and B.

It is assumed that every node carries a large stock of both types of pseudonyms which can be used for authentication. Additionally, every node is equipped with a node certificate which is signed by a trustworthy authority. The number of pseudonyms should be large enough that repeated authentication is possible—even against an adversary—without using the same pseudonym twice. This apparently requires mechanisms to limit the number of authentication attempts per time unit; such a limitation is not part of the protocol definition, but is strongly advised to be part of a protocol policy.

Authentication Sequence: We now introduce the authentication sequence displayed in Figure 2 and continue to use the notation from the previous section: Node A has received an encoded, but undecipherable frame from node B and decides to react on that frame.

1. **Pre-Authentication using Perfect Pseudonyms (PPs):** Node A starts a secret handshake with a not previously used Perfect Pseudonym. Appended to this is a block containing a publicly known value and random padding, encrypted with the current MANET key. The PP does not reveal any information about the node's identity or the node's group identity (CA). Details about the three-message handshake sequence can be found in [8]. The appended block prevents other nodes from network $\mathcal{A}$, which are also able to decode that block, to answer the handshake challenge. B can not decrypt the block, and this will act as a trigger to answer the challenge. Except for the encrypted block, the message is clear text, with the sender address made anonymous, and is transmitted via broadcast since the address of B is still unknown.

2. As a result of the PP handshake, both nodes share a common key that remains unknown to eavesdroppers. Using this key, they can build a point-to-point communication channel between them that is pre-authenticated and secured against all other nodes. We call this communication channel a *bridge* and the temporary common key of A and B the *bridge key*. For all frames using that bridge key, the *Bridge Flag* is set, so that other nodes will ignore those messages.

   But since the PPs cannot easily be revoked, one of the nodes could have been compromised and may still be able to initiate the bridge. Therefore, we continue with a second authentication step.

3. **Authentication using Group-identifiable Pseudonyms (GiPs):** Now, node B broadcasts one of its not previously used Group-identifiable Pseudonyms, GiP^i_B, together with a nonce n_B. This message is encrypted using the bridge key.

4. Node A now checks the validity of the signature attached to GiP_B^i using its copy of the CA public key. A then checks GiP_B^i against its revocation list by trying to decrypt the blob and getting the publicly known value. Then, A extracts the public key Pub_B^i from GiP_B^i. Node A chooses an own pseudonym GiP_A^j and encrypts the nonce n_B using the secret private key Priv_A^j extracted from GiP_A^j:
$$V_A(n_B) = \mathrm{Enc}_{\mathrm{Priv}_A^j}(n_B)$$
It then generates a nonce n_A and sends a message containing $\mathrm{GiP}_A^j, V_A(n_B), n_A$ to node B.
5. Node B now checks the validity of GiP_A^j using the signature, checks the encrypted blob in GiP_A^j against its revocation list and extracts the public key Pub_A^j from GiP_A^j. It then proves that
$$\mathrm{Dec}_{\mathrm{Pub}_A^j}(V_A(n_B)) \stackrel{?}{=} n_B \quad .$$
If this succeeds, B fully trusts A. Node B now calculates
$$V_B(n_A) = \mathrm{Enc}_{\mathrm{Priv}_B^i}(n_A)$$
and sends this value to A.
6. When node A receives that message, it can check that
$$\mathrm{Dec}_{\mathrm{Pub}_B^i}(V_B(n_A)) \stackrel{?}{=} n_A \quad .$$
If that succeeds, A fully trusts B.

We now have a bridge between two nodes that have checked their general authentication, checked their group authentication and trust each other.

4.5 Transparent Bridge Usage

When nodes A and B have authenticated each other and the secured communication bridge has been established nodes can communicate with each other. But their secret bridge key is not known to their respective networks $\mathcal{A}$ and $\mathcal{B}$. Frames sent over the bridge will be ignored by other nodes since the Bridge Flag is set. As long as there is no common MANET key, nodes A and B will be responsible to transparently forward traffic over the bridge:

- For every broadcast message node A receives from network $\mathcal{A}$, it will forward this message twice: once using the MANET key and once using the bridge key. Node B will then re-broadcast this message with its MANET key into $\mathcal{B}$. This forwarding explicitly includes management frames from the MANET routing protocol. The same rule applies for node B forwarding messages in network $\mathcal{B}$ and to node A.
- Unicast messages received by A from a node in $\mathcal{A}$ will be forwarded once according to any forwarding rules by the MANET routing protocol. If the next hop is B the message will be forwarded to B using the bridge key. If the next hop is in $\mathcal{A}$ the message will be forwarded using the MANET key of $\mathcal{A}$. Here, too, the same rule applies for node B and network $\mathcal{B}$ with respect to node A.
- Depending on the next hops to be reached, multicast messages must be forwarded like unicast or broadcast messages.

With this transparent bridge, any route discovery mechanisms in the routing protocol will soon integrate nodes from $\mathcal{A}$ and $\mathcal{B}$. We can thus gain a complete connectivity between the two networks even before a common MANET key is established, which can dramatically increase the network reactivity in case of topology changes. This approach also works if multiple bridges exist between networks $\mathcal{A}$ and $\mathcal{B}$ simultaneously. In this case the routing protocol can even find better routes between nodes in separate networks and the load on single bridges can be reduced.

To finally merge the networks $\mathcal{A}$ and $\mathcal{B}$, a common MANET key for all nodes has to be established.

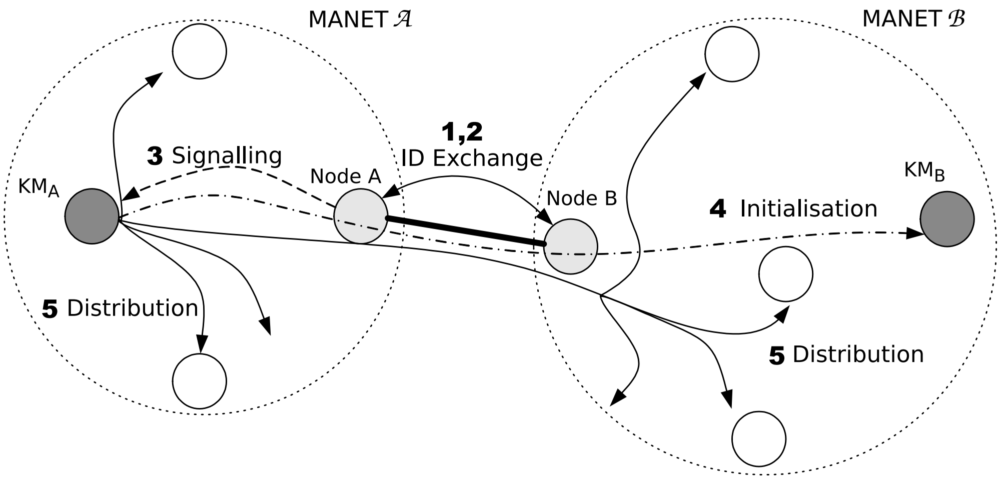

Figure 3: Key management consolidation between two secured networks using a temporary bridge.

4.6 Key Management Consolidation

We assume that the key management protocol used in SEAMAN offers a single Key Management (KM) node in every network. This node is responsible for key distribution and key updates within the network. The role of the KM may be taken by any node within the network, and this association can change over time, e. g. due to topology changes. A solitary node always takes the role of a KM and forms a single-node network.

When a new node joins a network or two networks merge, one of the two KM nodes involved has to be informed of the change to initiate a MANET key update on all nodes. Five steps are necessary for key consolidation (see Figure 3):

1. B sends its own ID (the public part of the certificate) and the ID of its Key Management (KM) node to A, encrypted using their common *bridge key*.
2. A sends its own ID and the ID of its KM to B, also encrypted with the bridge key.
3. Both A and B compare the ID of their own KM and that of the foreign KM. The node whose KM has the lower ID sends a *Merge* notification message to its own KM. Since the IDs of KMs must be unique, e. g. based on their IP address, A and B should not continue if both IDs are equal.

   W. l. o. g. we assume that the Key Management node $\mathrm{KM}_{\mathcal{A}}$ of network $\mathcal{A}$ has the lower ID. So, A sends the *Merge* notification to $\mathrm{KM}_{\mathcal{A}}$.
4. $\mathrm{KM}_{\mathcal{A}}$ will initialise a connection to $\mathrm{KM}_{\mathcal{B}}$ to inform it of the merge.
5. Then, $\mathrm{KM}_{\mathcal{A}}$ will distribute a new MANET key to all nodes. The details of this procedure are not in the scope of SEAMAN. The two KMs may need to authenticate each other during their communication as a protection against internal attackers, but this is not mandated by SEAMAN.

4.7 MANET Key Update

Whenever a new MANET key is distributed in both networks, all nodes have to switch their link layer encryption to use the new key. Because the key distribution can take an arbitrary amount of time and synchronised clocks are not required in the MANET, it is not possible to switch the key on all nodes simultaneously at a defined point in time. Instead, a transition period of time is allowed between the old and the new key where both keys are valid. The transition is performed as follows (figure 4):

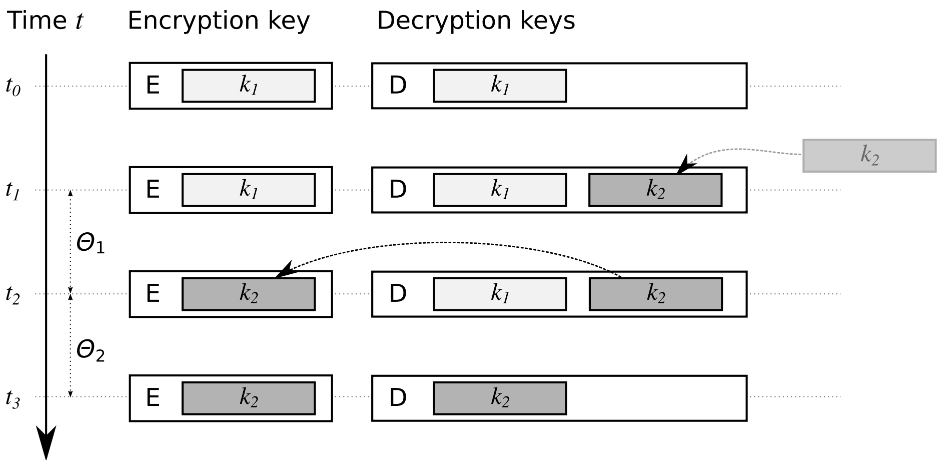

Figure 4: Key update sequence on a MANET node.

1. At time t_0, node X uses a single MANET key k_1 for decryption of incoming messages and encryption of outgoing messages.
2. At time t_1, X receives a new MANET key k_2 and informs its MANET routing process of this key. k_2 is immediately added to the list of keys valid for decryption.
3. At time $t_2 = t_1 + \Theta_1$, key k_2 replaces key k_1 as the key used for encryption of outgoing messages.
4. At time $t_3 = t_2 + \Theta_2$, key k_1 is removed from the list of keys valid for decryption. From now on, only k_2 is a valid MANET key.

The timeout values Θ_1 and Θ_2 should be adapted to the maximum time difference expected for propagation of key updates to a node and its neighbour nodes, but it should be as short as possible to ensure forward and backward secrecy in the network. This timing can be optimised if the MANET routing protocol supports neighbourhood discovery. Signalling can then be used to announce the availability of the new key k_2 to all neighbours. A neighbour can decide to use key k_2 for encryption as soon as all neighbours announced the *availability* of k_2. Also, a node can discard the old key k_1 as soon as all neighbours *use* the new key k_2 in their messages. The concept described here can easily be extended to support multiple overlapping keys, caused by rapid key updates due to node ejection, for example.

4.8 Bridge Decomposition

The bridge between nodes A and B will eventually be decomposed when the common MANET key is established. This can be detected by any of the bridge heads when it receives a SEAMAN frame from its counterpart that it can decrypt successfully with the MANET key. Also, the bridge is decomposed when a link loss between A and B occurs, e. g. signalled by the the routing protocol or by timeout. To decompose the bridge, a node deletes the bridge key and the trust information associated with the corresponding bridge head.

5.0 NETWORK OPERATIONS WITH SEAMAN

5.1 Node Join and Network Merging

In SEAMAN the *Node Join* operation is regarded as a special case of the *Network Merge* operation with at least one network having only one node. The network merging process is initiated by the foreigner detection mechanism and the bridge building. After the establishment of a bridge the networks are connected, but the connection is optimised by assigning a common MANET key. This is done in the Key Management Consolidation and Key Update phases. When the key update is finished the bridge is automatically removed by the bridge endpoints.

5.2 Node Leave and Network Splitting

The *Node Leave* operation is a special case of the network splitting operation. A network split does not need to be detected by the routing protocol. It is automatically triggered when key management tries to establish a new key due to a re-keying timeout. The network containing the KM detects that some nodes are not reachable anymore and the new network that does not contain the KM will elect a new KM. The routing protocol may inform the other nodes and/or the KM that a network part is not reachable anymore. However, the split operation should not be performed immediately, or else a short break of connectivity would lead to different keys in the network parts and a new merge operation had to be performed after the reconnect. As in the merge operation the old key remains valid for a certain time and overlaps with the new keys.

5.3 Node Eject

The *Node Eject* operation may be triggered from outside the network. It is performed whenever an authorised authority demands the removal of a node from the network. This may occur when a node is compromised. In this case the certificate of the node is revoked and the symmetric key for the GiPs is published. This will also trigger the key management to initiate a key update which will exclude the node. If the node tries to reconnect to the network, it may perform the first step of authorisation with the Perfect Pseudonyms, but cannot perform the authorisation with the Group-identifiable Pseudonyms. The ejected node may still be able to listen to the MANET traffic for a small time period when the old and new MANET keys overlap. Thus the time period for overlapping has to be kept small.

6.0 SECURITY CONSIDERATIONS

The security aspects of our protocol are discussed in this section. The two main aspects of security are the anonymity of MANET nodes and the encryption of the whole link layer with a dynamic key.

External anonymity is guaranteed in SEAMAN by the use of Perfect Pseudonyms in the authentication part of the protocol. No identity-related information like MAC addresses or group IDs are sent before a secured channel has been built up. Traffic obfuscation techniques can be applied to respond to traffic analysis attempts.

The key management subsystem establishes and maintains a common MANET key even if members join or leave the MANET. All data leaving a MANET node is encrypted with the MANET key, including the complete routing management traffic. Authenticity and integrity is ensured by the HMAC appended to every message. The MANET key is changed every time a member joins or leaves the MANET, so *forward secrecy* as well as *backward secrecy* is weakly guaranteed, barring the transition time for node updates. In contrast to static pre-shared keys, the SEAMAN MANET key is *dynamic*.

There is an effective mechanism in SEAMAN to react even on internal security threats. An internal attacker tries to be indistinguishable from an legitimate participant. For this reason an internal attacker is able to do everything a legal user is allowed to until it is exposed. In particular it is possible to reveal all identities and listen to the MANET traffic of other nodes. This caveat can only be removed by the introduction of session keys for every possible pairing of nodes. From our point of view this solution imposes too much load and management overhead to the MANET, especially in tactical environments. It also makes efficient multicast impossible. Detection of internal attackers can be provided by an Intrusion Detection System in an automated way. The reaction of SEAMAN on an internal attacker is to publish the compromised node's symmetric key, revoke its certificate and update the MANET key. Now it is impossible for an internal attacker or a compromised node to reveal the identities of other nodes. It can still successfully pass the Perfect Pseudonym challenge, but it can only reveal the identity of the CA that was used for GiP signing which was most probably already known to the attacker. Also the adversary is permanently excluded from future MANET communication.

SEAMAN offers external anonymity, confidentiality, authenticity and integrity of both user traffic and routing messages, while providing effective mechanisms against internal attackers.

7.0 REALISATION

The protocol concept described in the above sections can be realised as a combination of the existing protocols WNet [10], MIKE [11] and MASK [8] with some minor enhancements. While WNet enables proactive routing within the network, MIKE provides a mechanism which allows to dynamically change the MANET group key and MASK includes the mechanism to perform perfectly anonymous authentication.

The WNet framework enables efficient uni- and multicast communication. It proactively determines the network topology based on HELLO and Topology Control (TC) messages, similar to the OLSR MANET protocol. Routing itself is performed using the Dijkstra algorithm with a link quality metric. Therefore the protocol is able to quickly adapt to the latest network topology changes. Applying a common MANET key to WNet nodes ensures that only nodes with the same key are able to communicate with each other. An enhanced WNet variant supporting transparent bridges could automatically merge two network topologies as soon as the bridge would be established.

We decided to choose MIKE as the key management protocol because the key establishment algorithm is resilient against various kinds of network failures. Significant transmission capacity is saved by the utilisation of multicast. The common key established by MIKE is not bound to a specific use and can therefore be used for link layer encryption. After the common MANET key is established by MIKE, it is used by WNet for symmetric encryption of all MANET traffic.

8.0 CONCLUSION

Our SEAMAN protocol concept addresses the aspects and challenges mentioned. We propose to combine a MANET routing protocol with a key management system to establish secure routing. The MANET key supplied by the key management system is used to encrypt all traffic within the network. The main challenge is to dynamically react to network operations like *Merge* and *Split*. Once two separate MANETs get into radio range of each other, a temporary bridge between these two networks is built up, ensuring anonymity and authenticity of the peers. The key management system then distributes a new MANET key in both interconnected networks. The two networks are merged by establishing the MANET key at every node. Overlapping key schedules provide connectivity without interruption.

Our concept evades the chicken-and-egg problem of secure routing: On the one hand it is not safe to deliver a secret key over an unsecured network. On the other hand, if the network is encrypted, a new node cannot take part in routing and therefore is not capable to receive the secret key in a secure way. By introducing a temporary anonymous and authenticated connection the conflict can be solved.

REFERENCES

[1] NC3A, *NATO Network Enabled Capability Feasibility Study*, Vol. II, NATO Consultation, Command and Control Agency, 2.0 ed., 2005.

[2] Roy, S., Addada, V. G., Setia, S., and Jajodia, S., "Securing MAODV: Attacks and Countermeasures," *Proc. 2nd IEEE Intl. Conf. SECON*, IEEE, 2005.

[3] Kaya, T., Lin, G., Noubir, G., and Yilmaz, A., "Secure Multicast Groups on Ad Hoc Networks," *Proceedings of the 1st ACM Workshop on Security of Ad Hoc and Sensor Networks Conference*, 2003.

[4] Galera, F. J., Ruiz, P. M., Gomez-Skarmeta, A. F., and Kassler, A., "Security Extensions to MMARP Through Cryptographically Generated Addresses," *Lecture Notes on Informatics*, Vol. P-68, 2005, pp. 339–343.

[5] Curtmola, R. and Nita-Rotaru, C., "BSMR: Byzantine-Resilient Secure Multicast Routing in Multi-hop Wireless Networks," *Proceedings of the 4th Anual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, 2007.

[6] Kong, J. and Hong, X., "ANODR: ANonymous On Demand Routing with Untraceable Routes for Mobile Ad-hoc Networks," *Proceedings of the 4th ACM International Symposium on Mobile Ad hoc Networking and Computing*, 2003.

[7] Zhu, B., Wan, Z., Kankanhalli, M. S., Bao, F., and Deng, R. H., "Anonymous Secure Routing in Mobile Ad-Hoc Networks," *LCN '04: Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks*, IEEE Computer Society, Washington, DC, USA, 2004, pp. 102–108.

[8] Zhang, Y., Liu, W., Lou, W., and Fang, Y., "MASK: Anonymous On-Demand Routing in Mobile Ad Hoc Networks," *IEEE Transactions on Wireless Communications, vol.5, n^{o}9.*, 2006.

[9] Argyroudis, P. G. and O'Mahony, D., "Secure Routing for Mobile Ad hoc Networks," *IEEE Communications Surveys and Tutorials*, Vol. 7, No. 3, 2005, pp. 2–21.

[10] Bachran, T., Bongartz, H. H.-J., and Tiderko, A., "A Framework for Multicast and Quality based Forwarding in MANETs," *CCN '05: Proceedings of the 3rd IASTED International Conference on Communications and Computer Networks*, ACTA Press, 2005, pp. 120–125.

[11] Aurisch, T., "Optimization technique for military multicast key management," *Unclassified Proceedings of the IEEE MILCOM*, IEEE, Atlantic City, 2005.