Institut Eurécom
Department of Corporate Communications
2229, route des Crètes
B.P. 193
06904 Sophia-Antipolis
FRANCE

Research Report RR-09-233

Bootstrapping security associations in content-based opportunistic networks

15/06/2009
Revised 20/01/2010

Abdullatif Shikfa, Melek Önen and Refik Molva

Tel : (+33) 4 93 00 26 26
Fax : (+33) 4 93 00 26 27
Email : {abdullatif.shikfa, melek.onen, refik.molva}@eurecom.fr

[1]Institut Eurécom's research is partially supported by its industrial members: BMW Group Company, Bouygues Téécom, Bouygues Telecom, Cisco Systems, ThalesFrance Téécom, Hitachi, SFR, Sharp, ST Microelectronics, Sharp, Swisscom, Texas Instruments, Thales.

Abstract

Key management in opportunistic networks is a challenging problem that cannot be solved with existing solutions. In this paper, we analyze the requirements of key management in the framework of opportunistic networks and content-based forwarding. We then present a specific key management scheme that enables the bootstrapping of local, topology-dependent security associations between a node and its neighbors along with the discovery of the neighborhood topology, thanks to the use of pseudonym certificates and encapsulated signatures. This key management solution relies on two phases: a first phase where nodes are connected to an Identity Manager that provides them with unique pseudonyms to prevent Sybil attacks, and a second phase where the opportunistic communication and the security associations bootstrapping take place without the need for the Identity Manager. This solution with an offline Identity Manager is well-suited to opportunistic networks and can be used as an anchor to provide end-to-end confidentiality based on local and self-organized key management.

1 Introduction

Opportunistic networking ([10, 12]) is a new paradigm aiming at enabling communication through highly heterogeneous networks using different communication technologies. The delay-tolerant paradigm is a suitable approach to address the lack of connectivity and the mobility akin to opportunistic networks. In opportunistic networks, mobility and disconnections are the rule rather than the exception, therefore opportunistic networks are delay-tolerant by nature. The lack of end-to-end connectivity is a key difference between such networks and Mobile Ad-Hoc Networks (MANETs). This major constraint implies that it is impossible to establish an end-to-end path from source to destination and forwarding decisions are only based on a local view of the network.

Furthermore, opportunistic networks are more general than MANETs, because disseminational communication is the rule rather than conversational communication. A concept that nicely fits with the disseminational networking model is offered by content-based communication ([6, 7]) whereby messages are forwarded from source to destinations based on their content instead of explicit addresses. In content-based applications nodes declare their interests through receiver advertisements and simply publish content that they wish to disseminate, rather than explicitly defining destination nodes for packets. Intermediate nodes set up and update their forwarding table based on the receiver advertisements, and take forwarding decisions implicitly by looking up published content in their forwarding table.

The flexibility of content-based opportunistic networks come on the other hand with a high cost in increased exposure in terms of data security. Security services and in particular key management should be revisited to reflect the characteristics of such networks; in particular security services should also be flexible and self-organized. Moreover, privacy protection is particularly challenging due to the content-based messaging paradigm. The protection of the content with classical security mechanisms would indeed conflict with the forwarding functions since the latter rely on the very content that is being transmitted for their basic operations. An interesting idea to meet the privacy requirements of content-based forwarding in opportunistic networks consists of multiple layer commutative encryption (MLCE) that allows to perform secure operations on encrypted content as proposed in [19, 20]. When using MLCE, one needs to encrypt the data with several layers of encryption corresponding to r next hops. Such a solution therefore calls for an innovative key management scheme that should ensure local and self-organized security associations between a node and its neighborhood: each node should share a key with all its neighbors that are less than r hops away. The key management should thus depend heavily on the neighborhood topology which is fundamental for the multi-layer encryption scheme to work properly. Because of the lack of infrastructure, this also means that the neighborhood topology itself should be securely discovered.

The main goal of our work is therefore to propose a local, self-organized and topology-dependent bootstrapping of security associations along with a secure neighborhood discovery. In order to optimize the performance of the scheme, and to cope with the dependency between topology and security, it is indeed more efficient to perform both neighborhood discovery and security associations with all r-hops neighbors together rather than in two separate steps. We achieve this goal by using an authenticated version of Diffie-Hellman key agreement together with encapsulated signatures that protect the integrity of key management messages at each hop. Moreover, since the security of MLCE is directly linked to the number of consecutive colluding nodes, it is important to guarantee that each node can claim only one identity and only one position in the neighborhood. Creation of bogus identities through Sybil attacks would then be a crucial threat against which our scheme is protected thanks to the introduction of an off-line Identity Manager.

In this paper, we first analyze the new security challenges regarding key management in the context of opportunistic networks and extract important requirements for key management in this context. We then present a self-organized and local mechanism that bootstraps security associations with the discovery of the neighborhood topology thanks to the use of certificates and signatures chains. The proposed scheme relies on two phases: a first step where nodes are connected to an Identity Manager that provides them with unique pseudonyms, and a second step where the opportunistic communication takes place and where there is no need for the Identity Manager. The pseudonyms are not used as certified identities but only serve the purpose of withstanding Sybil attacks.

2 Problem statement

2.1 Privacy in content-based opportunistic networks

As mentioned in the introduction, content-based forwarding solutions raise entirely new privacy concerns: since nodes may not want to reveal the content of packets to entities other than destination(s), forwarding decisions should be taken over encrypted information. In [19], Shikfa et al. propose an interesting approach to meet the conflicting requirements between forwarding and privacy in content-based opportunistic networks. The idea of this approach is to use multiple commutative encryption layers in order to ensure end-to-end confidentiality: packets are encrypted with multiple keys where each of them is shared by a different pair of nodes. Thanks to this scheme, intermediate nodes securely compare published content and encrypted interests on the fly.

Even though it is impossible to establish an end-to-end path between source and destination, nodes can determine the r next hops with a local knowledge of the network. Each node establishes a secure channel with nodes that are r hops away. Moreover, the proposed scheme is commutative in the sense that $\{\{m\}_{k_1}\}_{k_2} = \{\{m\}_{k_2}\}_{k_1}$, thus layers can be removed in any order.

While sending a new packet, the source first encrypts it r times with r different keys, each of them being shared with one of the r next hops. Thanks to the commutativity of the encryption scheme, whenever an intermediate node receives an encrypted packet, it first removes one layer of encryption, and then compares the encrypted form of receiver advertisements and published content to take a forwarding decision. Before forwarding the packet, in order to ensure the confidentiality at the same level, the same intermediate node adds another encryption layer using the key that is shared with its rth next hop. By removing old encryption layers and replacing them with the new ones, the same confidentiality degree is always ensured based on only a local knowledge of the network. The security of the scheme of course strongly depends on this r parameter: the content of a packet can only be discovered if r consecutive nodes collude.

In order to illustrate this scheme, we define a simple network with 5 nodes and we set $r = 2$. Whenever Node 1 wishes to send a receiver advertisement raising its interest on a keyword w, w is encrypted both with k_{12} and k_{13}. When Node 2 receives this packet, it removes one encryption layer using k_{12}, updates its forwarding tabled with the partial information and finally adds another encryption layer using k_{24} before sending it to Node 3. Each node follows the same procedure. When Node 5 publishes some content with the same keyword w, the transitive and commutative properties of MLCE allow intermediate nodes to correctly forward this packet to Node 1. This scenario is summarized in figure 1.

In [19, 20] where MLCE is proposed and described, the problem of key management is overlooked. We address this problem by analyzing first the general requirements of key management in the context of opportunistic networks and then the more specific requirements of topology-dependent key management, in order to come up with a complete solution dedicated to MLCE.

2.2 Key management: Requirements and threats

Opportunistic networks have specific characteristics that raise new constraints and challenges which make existing key management protocols inefficient. In this section, we define the security specifications of our key management protocol.

On the one hand, the lack of end-to-end connectivity has strong implications on the problem of key management as well. Indeed, nodes cannot establish end-to-end security associations nor rely on an online, centralized authority or security server. Even though public key cryptography can be used with the restriction that trusted authorities may be unreachable or reply after a long delay, end-to-end public key encryption would still require a preliminary phase for the distribution of the destination node's certificate. Although identity-based cryptographic tools would be a good candidate for opportunistic networks because they do not require certificates (and they are used by Asokan et al. in [1] in this context), they are not suitable for content-based forwarding. Indeed, in content-based forwarding messages are forwarded depending on their content and the interests advertised by nodes, therefore the (set of) destination is unknown at the source.

On the other hand, the security of MLCE strongly depends on the location of the nodes in the topology. Indeed, nodes need to establish pairwise keys with all

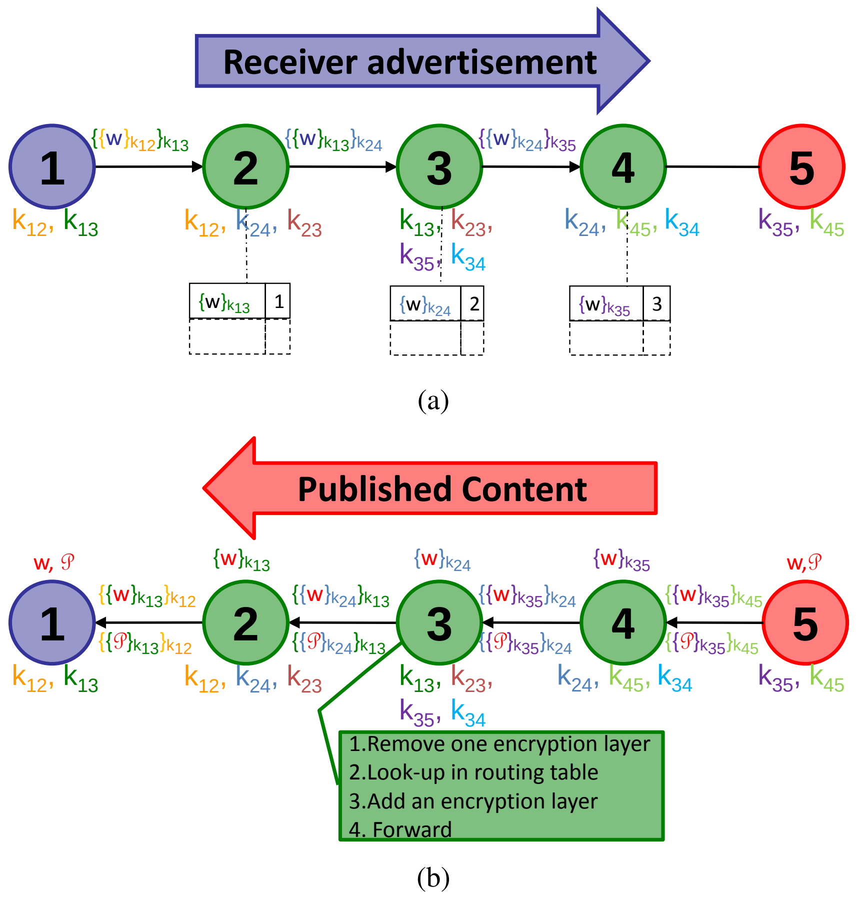

Figure 1: Multiple Layer Commutative Encryption overview: example with five nodes and $r = 2$. Each node shares keys with its one and two hops neighbors (shown below the nodes). (a)Receiver advertisement propagation: receiver advertisements are keywords that encrypted twice according to the keys shared with the next two hops. Intermediate nodes remove one encryption layer, build their routing tables with partially encrypted data, encrypt it again and forward it to the next hop. (b) Published content dissemination: the published content is also encrypted twice with the keys corresponding to the next two hops. The payload and the keyword corresponding to the content are encrypted separetly. Intermediate nodes can remove one encryption layer, look-up the result in their forwarding table, then they add an encryption layer and forward the packet to the next hop.

nodes that are at most r hops away. Given the layered structure, the assurance of privacy strongly depends on the position of the nodes in terms of hop-distance: the key agreement scheme should therefore depend on the topology of the neighborhood which needs to be securely discovered because of the lack of infrastructure. Securely discovering the neighborhood topology in turn requires security services because nodes should guarantee their claimed hop-distance to their neighbors and should not claim fake distances which would have an impact on the security of MLCE. In order to take into account the dependency between network topology and security, and in order to avoid running two separate protocols, one for neighborhood discovery and one for local key management, security associations should be locally bootstrapped along with a lightweight neighborhood discovery solution.

Moreover, nodes can easily launch Sybil attacks [8] by simulating many different identities claiming different hop distances. In this case, one single node (the malicious node) simulating r identities and claiming different positions for each identity would receive one key per layer and would therefore easily decrypt the content of packets although it does not have the right to. Hence, a node should only have a unique unspoofable identity (pseudonym) and thus, a global mechanism of identity management has to be defined.

Furthermore, as with the design of any communication protocol, the key management protocol should consider the regular attacks which can be classified as follows:

- **Passive attacks**: malicious nodes only eavesdrop on communication; they do not take part in the forwarding process and therefore can only discover the content of the packets if those are not protected. Therefore protocol messages should be encrypted.
- **Active attacks**: malicious nodes can either modify packets or launch replay or man-in-the-middle attacks. In the particular case of key management in MLCE, the goal of active attackers would be to discover a key by establishing security associations with a legitimate node without complying with the local topology. Pollution or other kind of attacks where nodes only aim at disrupting the protocol without gaining any advantage, are out of scope of this paper.

To summarize, content-based opportunistic networking requires a local and self-organized key management mechanism. Nodes should establish key pairs with all nodes which are at most r hops away. Moreover, nodes should also be able to recognize the position of each node in order to achieve the security goals of MLCE, and therefore security associations should be bootstrapped along with neighborhood discovery. Finally, as with any regular protocol, the new key management protocol should be protected from regular network attacks.

3 Our solution

In order to meet the requirements detailed in the previous section , we propose a solution for bootstrapping security associations which features two phases. Indeed, nodes require anchors to be uniquely identified in the network, and each node should have only one valid anchor to prevent Sybil attacks. Therefore, we propose first a setup phase, during which nodes are connected to an Identity Manager (IM) that generates and distributes these anchors in the form of certificates. The keying material received during this phase can be considered as long-term keying material that allows the computation of short-term keys resulting from the establishment of security associations in a secure way.

During the regular network operations, nodes do not need to communicate with the Identity Manager anymore and the long term keys are not used by the application. We hereafter describe these two phases in detail.

3.1 Setup phase

During the setup phase, nodes contact an IM, which is a lightweight security server that generates pseudonyms and certificates on-the-fly but does not manage certificates as in classical public key infrastructures. For the sake of clarity, we assume the existence of a single Identity Manager (IM), but the infrastructure could be more sophisticated with a distributed architecture for example. The IM generates a public/private key pair pk_{IM}/sk_{IM}, and pk_{IM} is known by all nodes. The role of the IM is twofolds:

1. **Enforcing privacy:** The IM first provides nodes with pseudonyms in order to enforce privacy. In opportunistic networks real identities are meaningless because most of the nodes which are encountered by a given node N_i are unknown to N_i. Hence, using actual identities only incurs a privacy threat with no additional advantage over pseudonyms.

2. **Prevention of Sybil attacks:** The IM links the pseudonym to a real identity and a public/private key pair and certifies it. Indeed, even though identities are meaningless, nodes should be restrained to a unique pseudonym otherwise they could have several identities, which would lead to Sybil attacks. If a node could impersonate other nodes or simply produce several identities for himself, it could pretend to be at several positions at the same time, and therefore break the multi-layer scheme.

To fulfill these tasks, each node N_i first generates a public/private key pair pk_i/sk_i and then sends pk_i to the IM. The IM first verifies that N_i owns the associated private key with a challenge-response exchange, and then requests the node for some information I_i to uniquely identify N_i. The requested set of information remains the same for all nodes at anytime (e.g. full name, date and place of birth) and is thoroughly verified by the IM (with the help of official documents like ID card or passport for example). The IM uses this set of information I_i together with a master key K (known only by the IM) in a message authentication code (MAC) function to generate a pseudonym for the node:

$$\mathcal{P}_i = MAC(I_i, K).$$

The IM then provides N_i with a certificate $\mathcal{C}_i$ which links the public key of N_i with its pseudonym, by signing these information:

$$\mathcal{C}_i = \{\mathcal{P}_i, pk_i, signature_{sk_{IM}}(\mathcal{P}_i, pk_i)\}.$$

The information exchange protocol between the IM and a node N_i is presented in figure 2. Note that a node can obtain several certificates with different public keys, but all the certificates include the same pseudonym and can therefore not be used for Sybil attacks.

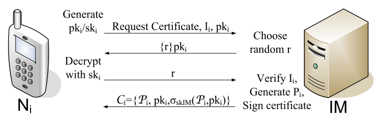
Generate pk_i/sk_i
Request Certificate, I_i, pk_i
Choose random r
$\{r\}pk_i$
Decrypt with sk_i
r
Verify I_i, Generate P_i, Sign certificate
$C_i=\{\mathcal{P}_i, pk_i, \sigma_{skIM}(\mathcal{P}_i, pk_i)\}$
N_i
IM

Figure 2: Summary of the information exchange protocol with the IM.

This certification process, hence, ensures that each node has only one pseudonym, and the certificate can be used to prove that this pseudonym was generated by the IM and is not random. Therefore, the use of this certificate effectively prevents Sybil attacks.

When the node N_i has retrieved its certificate $\mathcal{C}_i$, the setup phase ends and N_i can enter the runtime phase. During the runtime phase, communication is supposed to be delay-tolerant, therefore the IM is unreachable and secure communication should be possible without accessing the IM.

3.2 Bootstrapping local security associations

We now assume that all nodes have already performed the setup phase and own at least one certificate as mentioned in the previous section.

During this phase nodes need to establish ephemeral security associations with all their neighbors which are at distance less than r hops. As mentioned previously, this key agreement depends on the local topology and therefore requires a secure neighborhood discovery. In order to optimize the number of message exchanges and to cope with the dependency between security and topology, we propose a local key agreement protocol along with neighborhood discovery: one protocol run provides the initiator with both a correct view of its neighborhood topology at r hops distance and shared secrets with all r-hops or less neighbors in a batch. On the one hand, the neighborhood discovery mechanism is inspired by secure routing protocols (like [11]) with the noticeable difference that our solution is based on a hop count limit instead of targeting a destination: it therefore relies on signatures chains to guarantee the integrity of the discovered topology. Contrary to secure routing in MANET, the goal of our protocol is not to perform end-to-end secure routing which is irrelevant in opportunistic networks, but simply to discover the local topology of the network. On the other hand, the key agreement scheme is derived from an authenticated version of Diffie-Hellman key agreement protocol, also called the station to station protocol [22]. We therefore assume that all nodes know a group G with generator g suitable for a Diffie-Hellman protocol. Furthermore, all exponentiations are taken modulo the cardinal of the group $|G|$ and we do not mention this modular extraction in the sequel of the paper for the sake of clarity.

The protocol features four main steps. First a node initiates a Security Association Request for r hops, this request is then forwarded to neighbors until the r-th hop receives it. Then, a Security Association Reply is sent to the initiator through the reverse path of the request and finally the initiator can compute the shared keys. These four steps are detailed hereafter and an example of the execution of the protocol over one path is given in table 1.

3.2.1 Initiation of Security Association Request

When a node N_s wants to establish security associations with its neighbors, at distance less than r hops, it needs to initiate a Security Association Request. It first computes its Diffie-Hellman share g^{r_s} in order to establish short term keys with each of the neighbors. In order to prevent impersonation, N_s should also send its certificate received from IM during the previous phase. Finally, since the neighborhood discovery message should not be forwarded after the r-th hop, an additional iterator should be included in the message and should be decremented at each hop. N_s signs all these information to prove their authenticity and broadcast the following message:

$$< SARq, r, \mathcal{C}_s, g^{r_s}, \sigma_s > .$$

$SARq$ is just an identifier standing for Security Association Request and σ_s is a signature of the whole message with the private key sk_s, to be more precise

$$\sigma_s = signature_{sk_s}(SARq, r, \mathcal{C}_s, g^{r_s}).$$

3.2.2 Processing and forwarding of Security Association Requests

Upon receiving a Security Association Request, an intermediate node N_i first verifies the authenticity of the initial message and then N_i processes the Security Association Request. In order to prove that it is on the path of the request and validate its hop distance, it builds on the received message by adding its certificate and by decrementing the iterator. It also generates its Diffie-Hellman share and includes it in the message, and signs the modified message: this produces a

Table 1: Example of Security Association bootstrapping. The initiator N_1 discovers its 3-hop neighborhood and establishes security associations with three nodes. The underlined font indicates changed message fields, relative to the previous message of the same type.

	N_1 **initiates Security Association Request**
N_1 $N_1 \rightarrow *$	chooses r_1 $\sigma_1 = signature_{sk_1}(SARq, 3, \{\mathcal{C}_1\}, \{g^{r_1}\}, \{\})$ $< SARq, 3, \{\mathcal{C}_1\}, \{g^{r_1}\}, \{\sigma_1\} >$
	Processing of Security Association Request by intermediate nodes
N_2 $N_2 \rightarrow *$ N_3 $N_3 \rightarrow *$ N_4	verifies σ_1 and chooses r_2 and ρ_2 $\sigma_2 = signature_{\underline{sk_2}}(SARq, \underline{2}, \{\mathcal{C}_1, \underline{\mathcal{C}_2}\}, \{g^{r_1}, \underline{g^{r_2}}\}, \{\underline{\sigma_1}\}, \underline{\rho_2})$ $< SARq, \underline{2}, \{\mathcal{C}_1, \underline{\mathcal{C}_2}\}, \{g^{r_1}, \underline{g^{r_2}}\}, \{\sigma_1, \underline{\sigma_2}\} >$ verifies σ_1 and chooses r_3 and ρ_3 $\sigma_3 = signature_{\underline{sk_3}}(SARq, \underline{1}, \{\mathcal{C}_1, \mathcal{C}_2, \underline{\mathcal{C}_3}\}, \{g^{r_1}, g^{r_2}, \underline{g^{r_3}}\}, \{\sigma_1, \underline{\sigma_2}\}, \underline{\rho_3})$ $< SARq, \underline{1}, \{\mathcal{C}_1, \mathcal{C}_2, \underline{\mathcal{C}_3}\}, \{g^{r_1}, g^{r_2}, \underline{g^{r_3}}\}, \{\sigma_1, \sigma_2, \underline{\sigma_3}\} >$ verifies σ_1 and chooses r_4 and ρ_4 $\sigma_4 = signature_{\underline{sk_4}}(SARq, \underline{0}, \{\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3, \underline{\mathcal{C}_4}\}, \{g^{r_1}, g^{r_2}, g^{r_3}, \underline{g^{r_4}}\}, \{\sigma_1, \sigma_2, \underline{\sigma_3}\}, \underline{\rho_4})$
	Beginning of Security Association Reply($remaining\_hop\_count = 0$)
$N_4 \rightarrow N_3$ $N_3 \rightarrow N_2$ $N_2 \rightarrow N_1$	$< SARp, \{\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3, \mathcal{C}_4\}, \{g^{r_1}, g^{r_2}, g^{r_3}, g^{r_4}\}, \{\sigma_1, \sigma_2, \sigma_3, \sigma_4\}, \{\rho_4\} >$ $< SARp, \{\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3, \mathcal{C}_4\}, \{g^{r_1}, g^{r_2}, g^{r_3}, g^{r_4}\}, \{\sigma_1, \sigma_2, \sigma_3, \sigma_4\}, \{\rho_4, \underline{\rho_3}\} >$ $< SARp, \{\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3, \mathcal{C}_4\}, \{g^{r_1}, g^{r_2}, g^{r_3}, g^{r_4}\}, \{\sigma_1, \sigma_2, \sigma_3, \sigma_4\}, \{\rho_4, \rho_3, \underline{\rho_2}\} >$
	Key computation
N_1	Verify the validity of the reply Shared keys : $g^{r_1 r_2}$ with N_2, $g^{r_1 r_3}$ with N_3 and $g^{r_1 r_4}$ with N_4 One established 3-hop path : N_2, N_3, N_4

sequence of encapsulated signatures which validates the integrity of the message at each step. Thus, the general form of a Security Association Request contains three lists gradually filled in by intermediate nodes:

$$< SARq, remaining\_hop\_count, Certificate\_list, \\ DH\_share\_list, signature\_list > .$$

To be more precise, N_i first checks the authenticity of the initial request message by verifying the signature of the initiator. To do so, it reconstructs the initial request message which is:

$$< SARq, r, first(Certificate\_list), first(DH\_share\_list), \\ first(signature\_list) >$$

where first(.) designates the first element in a list. r is computed as the addition of $remaining\_hop\_count$ and the number of elements in the lists minus one. Then, the initial signature $first(signature\_list)$ is checked thanks to the public key of the initiator which can be found in
$first(Certificate\_list)$.

If the signature is valid, the intermediate node N_i processes the request as follows:

- $remaining\_hop\_count$ is decreased by one,
- N_i appends its own certificate $\mathcal{C}_i$ to $Certificate\_list$ in order to give a proof of its pseudonym $\mathcal{P}_i$ and to provide its public key pk_i,
- N_i needs to provide a Diffie-Hellman share for the key agreement, hence N_i draws a random number r_i and then appends g^{r_i} to DH_share_list,
- N_i needs to prove the integrity and authenticity of the modified request therefore it computes a signature σ_i of the modified message plus a random number ρ_i:

  $$\sigma_i = signature_{sk_i}(ND, remaining\_hop\_count, \\ Certificate\_list, DH\_share\_list, \rho_i)$$

  and appends σ_i to $signature\_list$.

ρ_i is a random number that is revealed in the Security Association Reply as described in the next section. This random number guarantees that the reply returns through N_i: if the reply do not pass through N_i then σ_i cannot be verified and therefore the message is considered as not valid.

After this processing, the message is broadcasted, or a reply message is sent back to the initiator if the message reached the r-th hop.

3.2.3 Security Association Reply

The reply has to follow the reverse path from which the discovery request has been forwarded, therefore the iterator is no longer needed. The reply mainly consists of the list of certificates, signatures and Diffie-Hellman shares at the last hop of the request. Furthermore, intermediate nodes N_i that receive back the reply, need to reveal the random number ρ_i they used in the request to allow the verification of their signature. Therefore the general format of the reply is:

$$< SARp, Certificate\_list, DH\_share\_list, signature\_list, \\ random\_number\_list > .$$

$SARp$ is an identifier for the reply and $random\_number\_list$ corresponds to the list of random numbers used during the signatures of request messages.

The processing of reply messages by intermediate nodes is simple. Upon receiving a reply message, an intermediate node N_i first checks that it was on the request path, by looking for its own certificate $\mathcal{C}_i$ in $Certificate\_list$ and then appends the random number ρ_i it chose to $random\_number\_list$. Then N_i forwards the message to the next hop as listed in the $Certificate\_list$.

3.2.4 Key computation

When the reply finally gets back to the initiator of the neighborhood discovery N_s, N_s thoroughly verifies its validity by checking that:

1. the number of elements in $Certificate\_list$, DH_share_list, $signature\_list$ is equal to $r + 1$ while the number of elements of $random\_number\_list$ is equal to r,

2. all the certificates in $Certificate\_list$ are related to different users (the pseudonyms should all be different) and valid (the signature of the IM on each certificate should be valid),

3. all the signatures in $signature\_list$ are valid. To do so, the initiator reconstructs the message at each hop and verifies the validity of the signature at each step by taking into account the corresponding random number listed in $random\_number\_list$.

If all these verifications succeed, N_s and the neighbors listed in the message compute their shared keys. The key shared with N_i is computed as $(g^{r_i})^{r_s}$ by the initiator and as $(g^{r_s})^{r_i}$ by N_i. N_s also knows of one r-hop path in its neighborhood.

Note that, for one Security Association Request, the initiator should receive many replies, one per possible r-hop path. Thanks to this mechanism, the initiator can fully construct its r-hop neighborhood topology and establish security associations with all the nodes in this neighborhood.

3.3 Evaluation

In the previous sections, we presented a complete mechanism to bootstrap security associations along with neighborhood discovery in opportunistic networks. The proposed mechanism is local and self-organized and therefore complies with the delay-tolerant nature of opportunistic networks.

The mechanism relies on two phases: a setup phase where nodes have access to the IM and the runtime phase where the opportunistic communication actually takes place. The proposed IM has a completely different role than classical Certification authorities. The role of the IM is not to certify identities, it just certifies that a given node has one and only one pseudonym. The pseudonym itself has no significance and it is not used as an identity in further communications, its only role is to guarantee that a node cannot impersonate other nodes. Furthermore, the IM is lightweight by design because it does not need to keep track of the certificates it delivered. Each time a node asks for a certificate, the IM generates the associated pseudonym on-the-fly by requesting the same information, and the resulting pseudonym is always the same for the same node, therefore each node can only have one pseudonym. During networking operations, the Identity Manager is not required anymore and the proposed scheme enables local and self-organized security associations.

We now evaluate the security of the complete mechanism first against eavesdropping and then against active attackers.

Since the establishment of security associations is simply based on the Diffie-Hellman exchange protocol, eavesdropping is inherently prevented thanks to the hardness of the Discrete Logarithm and the Computational Diffie-Hellman Problems [22]. Indeed, since given g^{r_1}, it is difficult to retrieve r_1 and given g, g^{r_1}, g^{r_2} it is difficult to compute $g^{r_1 r_2}$, key shares can be sent in clear and an adversary node cannot discover the key resulting from the association. Therefore, the security of the scheme against passive attackers results directly from the security of the Diffie-Hellman protocol.

However, since the message exchange is not performed by only two nodes, the security guarantee offered by the Diffie-Hellman protocol is not sufficient, especially in the presence of active attackers. The first type of attacks that can be launched by an active attacker is man-in-the-middle attacks. Such attacks are effectively prevented by the use of an authenticated version of the Diffie-Hellman exchange protocol that adds signatures computed over key shares. Indeed, no node can forge a network discovery request initiated by node N_s because it requires the private key of N_s. An authentic request by N_S can still be replayed by a malicious node. To mitigate this issue the classical solution consists in adding a timestamp. The use of timestamps requires time synchronization though and this is difficult to provide in opportunistic networks. Anyway a malicious node which replays a neighborhood discovery request cannot share a key with other nodes because it does not know the random number r_s. Furthermore, since nodes still answers several identical requests by processing them the same way (and by using the same

Diffie-Hellman share), this does not create false security associations, therefore this attack is not critical from a security perspective.

In fact the mechanism of encapsulated signatures prevents most basic active attacks, and makes tampering of Security Association messages difficult:

- the mechanism of encapsulated signatures in security association requests protects the integrity of messages at each step. Therefore an intermediate node cannot forge the message of a previous node, in particular it cannot change the value of an iterator at a previous step, nor can it modify the value of the Diffie-Hellman share. An intermediate node can only undo some steps to remove some nodes from the path and extend the neighborhood discovery hops in a grayhole attempt. But in this case the deleted nodes will not accept to forward the reply because their certificates are not in the certificate list anymore. To be successful this attack thus requires a way to circumvent the deleted nodes and in this case it is a wormhole and not a grayhole attack anymore.

- the mechanism also ensures that the path of the reply is the reverse of the request thanks to the use of the random numbers ρ_i. Indeed the signatures in the request messages cannot be verified if the ρ_i are not revealed and nodes only reveal them in reply messages if they were involved in the request path. An alternate solution would be to sign all the reply messages, but this would be more costly.

Wormhole attacks that completely circumvent the deleted nodes and avoid message discarding can be successful and the source node would end up with a fake neighborhood topology in that it would contain nodes which are more than r-hops away. The impact of this attack is however the same as the collusion attack in MLCE: if r consecutive nodes collude they can break the scheme and access encrypted messages. Hence, it is possible to mitigate this attack by increasing the security parameter r, which is chosen according to the expected maximum number of consecutive malicious nodes. Furthermore, we assume that nodes can securely determine their one-hop neighbors by using distance bounding techniques ([5, 21]), which further mitigates the wormhole threat.

Furthermore, as previously explained, thanks to the initialization phase whereby nodes communicate with the Identity Management system in order to get identity certificates, the proposed mechanism is automatically protected against Sybil attacks. Indeed, since the pseudonym of a node is strongly linked with its real identity, malicious nodes cannot simulate multiple nodes and thus cannot access any private message they are not authorized to.

Finally, we briefly evaluate the performance of the scheme. The scheme requires asymmetric cryptography and signatures to guarantee the local neighborhood topology. Nevertheless, the design of the mechanism takes into account the need to minimize the number of signatures. The use of the random numbers ρ_i serves this purpose, since it avoids signing both requests and replies, and enables the signature of requests only. Therefore intermediate nodes have to verify and to compute only one signature each, while the initiator has to verify only r signatures. The message length is roughly the size of the three main lists $Certificate\_list$,DH_share_list, $signature\_list$ which contain at most $r + 1$ elements each, and in each of these elements the most important component has a size of 1024 bits. The message length is therefore linear in the number of hops r.

It is worth noticing that the proposed protocol is not used for routing, but to bootstrap security associations from scratch. The proposed scheme can therefore be used as an anchor for further efficient key management based on these security associations. Using asymmetric cryptography to bootstrap security associations is a widely accepted concept, hence performance is not a critical issue for the proposed mechanism.

4 Related work

The area of key management in opportunistic networking is quite new and the existing work in this area are rare: in [9], Farrell mentions some requirements of key management in DTN but no solution is proposed, and in [1] Asokan et al. evaluate ID-based cryptography in the context of DTN, but this solution is not suitable for content-based forwarding as mentioned in section 2.2. In the broader area of peer-to-peer key management in mobile ad hoc networks (MANETs) many solutions have been proposed ([18]). These solutions can be classified in two main categories:

- fully self-organized key management, which have been first proposed by Capkun et al. in [3], and further studied in [2, 4, 17]. These solutions require no authority, and are based on self-certificates (PGP-like) which are then used to sign other trusted nodes' certificates to form chains of trust. Key management therefore requires high-mobility to efficiently establish the chains of trust. Unfortunately, trust establishment is a time consuming operation. Furthermore, such fully organized schemes are inherently vulnerable against Sybil attacks, which is a major issue for MLCE (see section 2.2). Therefore fully self-organized key management cannot fit to our problem.

- authority-based solutions, rely on an external authority to bootstrap trust relations from certificates signed by the authority. In addition, most of them make use of an online authority with the accent on distributing this online authority either partially ([14, 23, 24, 25, 26]) or fully ([15, 16, 13]). All these approaches are based on threshold cryptography and require each certificate to be signed more than once online and therefore they are not suited to our problem either.

An important difference between all these solutions and our proposal is that key management in MANETs aims at establishing end-to-end keys whereas this is irrelevant in opportunistic networks. It is therefore hard to compare these solutions with ours, but we can tentatively say that our solution is in between the two mentioned categories: it makes use of an offline authority to prevent Sybil attacks, but online key agreement is self-organized and does not require an additional online authority, therefore it meets the DTN requirements.

5 Conclusion

The analysis of the characteristics of opportunistic networks and content-based forwarding, lead us to the conclusion that key management in such networks should be self-organized and local. This locality also involves a correct view of the neighborhood topology. We therefore designed a complete solution that enables bootstrapping of security associations along with secure neighborhood discovery.

This solution based on pseudonym certificates and encapsulated signature enables key agreement between a node (the initiator) and all its neighbors which are at distance less than r-hops without pre-established trust relationship or infrastructure. The solution also enables the discovery of the neighborhood's topology and withstands tampering by malicious nodes. We also proposed the use of an Identity Manager which provides each node with a unique certified pseudonym during a setup phase. This lightweight IM therefore effectively prevents Sybil attacks. Furthermore the IM is offline and is not required during networking operations; therefore the key management scheme is self-organized.

The proposed scheme can therefore be used as an anchor to content based forwarding in opportunistic networks based on multiple layer commutative encryption, which results in end-to-end confidentiality and privacy-preserving content-based forwarding solely based on a local and self-organized key management.

References

[1] N. Asokan, K. Kostiainen, P. Ginzboorg, J. Ott, and C. Luo. Towards securing disruption-tolerant networking. Technical Report NRC-TR-2007-007, March 2007.

[2] M. Cagalj, S. Capkun, and J. Hubaux. Key agreement in peer-to-peer wireless networks. *Proceedings of the IEEE*, February 2006.

[3] S. Capkun, L. Buttyán, and J.-P. Hubaux. Self-organized public-key management for mobile ad hoc networks. *IEEE Transactions on Mobile Computing*, 2003.

[4] S. Capkun, J. Hubaux, and L. Buttyán. Mobility helps peer-to-peer security. *IEEE Transactions on Mobile Computing*, January 2006.

[5] S. Capkun and J.-P. Hubaux. Secure positioning in wireless networks. *Selected Areas in Communications, IEEE Journal on*, 24, Feb. 2006.

[6] A. Carzaniga, M. J. Rutherford, and A. L. Wolf. A routing scheme for content-based networking. In *IEEE INFOCOM 2004*, Hong Kong, China, March 2004.

[7] A. Carzaniga and A. L. Wolf. Forwarding in a content-based network. In *SIGCOMM*, pages 163–174, 2003.

[8] J. R. Douceur. The sybil attack. In *IPTPS '01: Revised Papers from the First International Workshop on Peer-to-Peer Systems*. Springer-Verlag, 2002.

[9] S. Farrell. Dtn key management requirements, June 2007. http://www.ietf.org/internet-drafts/draft-farrell-dtnrg-km-00.txt.

[10] Haggle project, 2006. http://www.haggleproject.org/index.php.

[11] Y.-C. Hu, A. Perrig, and D. B. Johnson. Ariadne: a secure on-demand routing protocol for ad hoc networks. *Wireless Networks*, January 2005.

[12] P. Hui, A. Chaintreau, J. Scott, R. Gass, J. Crowcroft, and C. Diot. Pocket switched networks and human mobility in conference environments. In *WDTN '05: Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, 2005.

[13] D. Joshi, K. Namuduri, and R. Pendse. Secure, redundant, and fully distributed key management scheme for mobile ad hoc networks: an analysis. *EURASIP Journal on Wireless Communications and Networking*, September 2005.

[14] A. Khalili, J. Katz, and W. A. Arbaugh. Toward secure key distribution in truly ad-hoc networks. In *Proceedings of the 2003 IEEE Symposium on Applications and the Internet Workshops (SAINT'03 Workshops)*, 2003.

[15] J. Kong, P. Zerfos, H. Luo, S. Lu, and L. Zhang. Providing robust and ubiquitous security support for mobile ad hoc networks. In *ICNP '01: Proceedings of the IEEE 9th International Conference on Network Protocols*, 2001.

[16] H. Luo, P. Zerfos, J. Kong, S. Lu, and L. Zhang. Self-securing ad hoc wireless networks. In *Proceedings of the IEEE Seventh International Symposium on Computers and Communications (ISCC'02)*, 2002.

[17] J. McCune, A. Perrig, and M. Reiter. Seeing-is-believing: using camera phones for human-verifiable authentication. In *IEEE Symposium on Security and Privacy*, May 2005.

[18] J. V. D. Merwe, D. Dawoud, and S. McDonald. A survey on peer-to-peer key management for mobile ad hoc networks. *ACM Computing Surveys*, 2007.

[19] A. Shikfa, M. Önen, and R. Molva. Privacy in content-based opportunistic networks. In *Workshop on Opportunistic Networking (WON)*, 2009.

[20] A. Shikfa, M. Önen, and R. Molva. Privacy-preserving content-based publish/subscribe networks. In *IFIP SEC 2009, 24th International Information Security Conference, May 18-20, 2009, Pafos, Cyprus*, 2009.

[21] R. Shokri, M. Poturalski, G. Ravot, P. Papadimitratos, and J.-P. Hubaux. A practical secure neighbor verification protocol for wireless sensor networks. In *WiSec '09: Proceedings of the second ACM conference on Wireless network security*, 2009.

[22] D. R. Stinson. *Cryptography: theory and practice*. CRC Press, Boca Raton, Florida, 1995.

[23] B. Wu, J. Wu, E. B. Fernandez, M. Ilyas, and S. Magliveras. Secure and efficient key management in mobile ad hoc networks. *Journal of Network and Computer Applications*, 30(3), 2007.

[24] G. Xu and L. Iftode. Locality driven key management architecture for mobile ad-hoc networks. In *IEEE International Conference onMobile Ad-hoc and Sensor Systems*, Oct. 2004.

[25] S. Yi and R. Kravets. Key management for heterogeneous ad hoc wireless networks. In *ICNP '02: Proceedings of the 10th IEEE International Conference on Network Protocols*, 2002.

[26] L. Zhou and Z. J. Haas. Securing ad hoc networks. *IEEE Network Magazine*, 1999.