Building Blocks for Secure Communication in Ad-hoc Networks

Christian Schwingenschlögl and Marc-Philipp Horn
Institute of Communication Networks
Munich University of Technology, Arcisstr. 21, 80290 Munich, Germany
Ph:+49 89 289 25821, Email: schwinge@lkn.ei.tum.de
Web: www.lkn.ei.tum.de/~schwinge

Abstract
Security is one of the main concerns in modern networks. Given the recent success of wireless networks, there are plenty of possibilities to attack the networks themselves [14][24] or services [28] running on these networks. Current progress in research and growing interest from the industry let also expect more and more sophisticated Ad-hoc networking solutions. Compared with security problems in conventional wireless networks, Ad-hoc networks are exposed to additional risks. In addition to that a reasonable level of security seems to be more difficult to achieve in these networks due to their dynamic and distributed nature. This paper will give an overview about basic security problems in Ad-hoc networks and the research that has been conducted in related areas so far. It presents some algorithms that can be used as building blocks for security in Ad-hoc networks. Given the constraints faced by such networks, the paper also inspects its practicability in Ad-hoc environments.

1 Introduction
As seen in e.g. [3], there seem to be quite a lot of problems concerning network security in conventional, wired networks. Difficulties in network security increased with the advent of wireless networks, mainly due to the now much more vulnerable physical link. However, apart from this difference, conventional wireless networks are mostly similar to wired networks from a security point of view.

If we regard the architecture of a typical Ad-hoc network, security solutions face new difficulties:

- Dynamic Nature: The topology of Ad-hoc networks can change very fast, depending on the movement of the nodes. This results in regular changes of the available routes, changes in the reachability of different nodes and changing participants.

- No central entities: Due to the dynamic nature of the Ad-hoc network, the functionality of all nodes should be equal. Central servers are not recommended for most application scenarios because of the complete break-down of the service when the server becomes unreachable for any reason. Regarding security services, also the high physical vulnerability of a single node should be taken into account.

- Unidirectional Links: Due to different power and transmission ranges of the nodes, unidirectional links can occur.

- Constraints regarding:
  - Processing power: Most of the scenarios for Ad-hoc networks assume mobile devices with small processors.
  - Battery power: If battery powered devices are used, the transmission power and the processor utilization will directly affect the battery lifetime.
  - Bandwidth: Is a scarce resource in the wireless world. This is especially true for mobile Ad-hoc networks that have to cope with a lot of additional signaling information and also have to act as relay-stations for neighboring network nodes.

- Scalability: Due to the high demands on decentralization and self-organization also scalability is an issue. Straightforward security mechanisms can have a high negative impact on system scalability.

Very often, user-convenience or system-performance are used as arguments for limited use of security mechanisms in networks. Given the constraints above, these arguments are especially important for Ad-hoc networks.

2 Threats faced by Ad-hoc Networks
For basic network security, the following security services are defined by the IETF:

- Availability: To ensure availability of a server to authorized parties when needed.

- Confidentiality: Protection of transmitted data from passive attacks (e.g. eavesdropping)

- Integrity: Protection of transmitted data against alteration by unauthorized parties.

1conventional wireless networks here means wireless clients that communicate via a base station (e.g. GSM, IEEE802.11)

2i.e. it can be easy to gain physical access to a node and therefore to remove, compromise or destroy it
• Authentication: Assurance that each entity is the entity it claims to be.

• Non-repudiation: To prevent any party from denying a transmitted message. I.e. the receiver can prove that the alleged sender sent the message and vice versa.

Given the very different application scenarios of Ad-hoc networks, a wide range of security requirements exists. Application scenarios start with very basic things like e.g. communication between mobile telephones and headsets. More demanding scenarios involve multiparty communication e.g. participants at a conference, sharing presentation slides and other information with their laptops and PDAs. Other scenarios involve more security-critical issues like wireless payment. One can also imagine larger Ad-hoc networks, e.g. voice communication or also data-communication between cars. For application scenarios in the business-world wireless payment and the transmission and access of confidential data are the most demanding ones. In addition, also emergency and military application scenarios of Ad-hoc networks are very common. In these scenarios, also physical attacks against the network nodes have to be considered. This can be a very serious security incident as the attacker can get access to hard- and software known to the network and can possibly perform successful authentication, eavesdrop messages or inject arbitrary or malicious data into the network.

Another variable factor in the selection of the right security architecture for Ad-hoc networks is the transmission technique used. If the network is based on very short-range transmission like e.g. IrDA, there clearly is no immediate need for sophisticated security solutions. This does change given techniques like e.g. Bluetooth [1] (range approx. 10m) and becomes even more important with devices using e.g. IEEE802.11 [11] or Hiperlan2 [4] that have an approx. range of 200m. Radio techniques in emergency and military scenarios do have even wider transmission ranges. The danger, however, is not necessarily limited to the range of the Ad-hoc network. Ad-hoc gateways to the Internet are a common scenario.

3 Requirements for Security in Ad-hoc Networks

One of the main shortcomings of wireless networks is the "unprotected" wireless link. I.e. to successfully attack a wired network, the attacker has to get physical access. For "external attackers" this is usually relatively hard to achieve. If a wireless network is used, the attacker has physical access within the transmission range. This usually also includes areas outside the protective building. Moving from wireless to wireless Ad-hoc networks, the dynamic system architecture turns out to be one additional main security problem. Network nodes suddenly come into range or disappear and in some scenarios the possibility of compromised nodes is high. This leads to a new design goal for security in Ad-hoc networks: Robustness of the networks security against a limited number of compromised nodes. Algorithms and protocols that can help to achieve this goal are presented in the following sections.

4 Previous Work

There has been some work so far on this topic. Lidong Zhou and Zygmunt J. Haas presented ideas about secure routing and a key management service in their paper [30]. Also a short introduction into the concepts of threshold cryptography is given. Another interesting paper was written by Frank Stajano and Ross Anderson [27]. In their paper they present the secure transient association of a device with multiple serialised owners. Their approach has been inspired by biology, the paper is called "The Resurrecting Duckling."

5 Secret- and Function Sharing

The main motivation for the utilization of secret- and function-sharing techniques is the vulnerability of single network nodes in a lot of scenarios. These techniques are mainly addressing two risks:

• If security relevant functions are provided by only one server node, a simple denial of service attack on this node will effect the security of the whole network.

• Due to their low physical protection, Ad-hoc nodes have a higher risk to become compromised. Techniques that require a benign operation of all nodes are therefore inappropriate.

5.1 Secret Sharing

Secret sharing is defined as distribution of a secret $s$ over $n$ parties. To reconstruct the secret $s$, $t \leq n$ parties are necessary. $t-1$ malicious parties can not gain additional information about the secret $s$ when they combine their shares. Secret sharing techniques are especially suited for Ad-hoc networking scenarios because

• the secret is protected, even if an adversary has access to max. $t-1$ shares.

• the secret remains available, even if shares get lost or are temporarily unavailable for some other reason.

Let $n$ be the number of available shares, $t$ the minimum number of shares that are necessary to reconstruct the secret. $S$ is defined as the "secret space", i.e. the set of all possible secrets. A secret sharing protocol needs the following participants:

• A dealer who knows the secret and performs the splitting to get the secret-shares,

• shareholders who receive the secret-shares from the dealer and a
5.2 Function Sharing

The term function sharing was introduced in 1994 by De Santis, Desmedt, Frankel and Yung in [23]. It basically is an analogon to secret sharing: A backdoor is opened and a function can be calculated if partial results from at least \( t \) out of \( n \) participants are available. However, there is one important difference to secret sharing: the cryptographic function can be computed multiple times without reducing system security. The cryptographic function is not made public at any time, only partial results and the result for a given input. In secret sharing algorithms the secret is made available at least to the combiner and should not be used any more after that.

5.3 Verifiable Secret Sharing

Verifiable secret sharing is an enhancement of ordinary secret and function sharing protocols. Usually, a so called “witness” is created by the dealer for every shareholder. If necessary, the combiner can verify each share or each result from a partial function using the witness. Faulty shares can be detected with this technique and the faulty or malicious shareholders can be isolated. Verifiable Secret Sharing protocols are divided into interactive and non-interactive protocols. Interactive protocols have a separate communication protocol to verify the partial results, in non-interactive protocols the combiner knows the witnesses directly.

If we imagine a \((t,n)\) Secret Sharing protocol with e.g. \( t = 4 \) and \( n = 10 \) we can see the importance of the possibility to verify the shares: 210 possible combinations of parties that can create a valid signature (compute a valid result from their shares) are possible. If only 3 parties are compromised and deliver faulty shares, only 90 possible combinations are left. If the single shares can not be verified, the combiner has to try out different combinations until he finds a correct one. As the combination of shares is computationally expensive, malicious shareholders have an easy possibility to launch denial-of-service attacks if they can not be identified.

5.4 Proactive Secret Sharing

In proactive secret sharing protocols it is possible for the shareholders to refresh their shares within given time-intervals. This results in a substantial increase in security, especially for systems with long operating times. An adversary has to compromise at least \( t \) shareholders within two resharinng cycles to be successful. When a refresh is done, the “network secret” stays the same, only the shares are changed. With proactive secret sharing, the adversary can compromise all nodes of the network during the system lifetime without being successful. To get control over the network, the adversary has to compromise
more than $t - 1$ nodes during the time interval between two refresh cycles\(^7\).

6 Protocol Review

In this section, some protocols that can be useful for securing Ad-hoc networks are briefly characterized. RSA has been an industry standard for more than 20 years and can now be used without licensing fees\(^8\). For this reason the focus of this section is on RSA-based algorithms\(^9\).

The following paragraph gives the definitions of the most important features of such protocols:

- **Security**: A threshold cryptosystem is said to be secure if $t - 1$ compromised or faulty nodes are not able to fake a signature with a possibility of $1 - \epsilon, (\epsilon < 1)$.

- **Robustness**: A threshold cryptosystem is said to be robust if $t - 1$ compromised or faulty nodes are not able to forestall the generation of a signature.

- **Optimality**: A threshold cryptosystem is said to be optimal if it is robust for the maximum possible number $f = \lceil \frac{t-1}{2} \rceil$ of compromised or faulty nodes.

- **Efficiency**: A threshold cryptosystem is often said to be efficient, however, no common definition of this feature was found by the author. Efficiency is used when the cryptosystem behaves reasonable with regard to memory usage and computation time.

- **Interactivity**: A threshold cryptosystem is said to be interactive if the parties have to communicate to generate a signature. E.g. some threshold cryptosystems include interactive sub-protocols for verifiable secret sharing. Also proactive protocols are highly interactive. An example for a non-interactive protocol is the function sharing protocol in Figure 2.

- **Synchrony**: A threshold cryptosystem is said to be synchronous if it defines hard limits on the maximum delay for transmitting or processing of messages.

Since [12] efforts were made to distribute the capability to create electronic signatures over many parties. Existing secret sharing schemes, including the elegant scheme from Shamir [25]), could not be converted straightforward into function sharing schemes. The basic problem is the algebraic structure of the keyset $\mathbb{Z}_{\phi(N)}$, that forms an algebraic ring and not a body. $\phi(N) = (p - 1)(q - 1)$ is an even number, and so it is not granted that there is an inverse element for every quotient $(x_i - x_j)$ of the lagrange-interpolation equation. Additionally, $\phi(N)$ has to be kept secret.

In [13], an elegant method was shown to circumvent this problem. Unfortunately, the method was a heuristic one and a formal proof of its security was not done. One year later, in [17], this proof was done by Frankel and Desmedt. Some changes in the protocol had to be made for it and these changes required the interaction of the shareholders to generate the signature. So this protocol was not any more the interactionless function sharing model advantageous for Ad-hoc networks.

In [23] the threshold scheme from Frankel was changed and the interactions were removed. However, the size of the shares was now growing proportional with $n$, the number of participating parties. In [20], this protocol has been expanded with an interactive and a non-interactive verifiable secret sharing protocol to a robust scheme.

[18] shows another approach: A probabilistic model that provides proactive security but is only efficient for very small threshold schemes. Unfortunately, this approach also does not fulfill the optimality criterium. However, in the same year the same authors did introduce a realistic threshold scheme for RSA in [19]. In this publication, the heuristic scheme from [13] has been modified and an additional secret sharing layer was added. This layer made the proof of security possible.

[22] was published by Tal Rabin in 1998. It describes an elegant approach to convert the additive $(n, n)$ threshold scheme from [16] into an efficient, robust and optimal $(t, n)$ scheme. Also a proof of security was given. The same publication describes a proactive sub-protocol.

Victor Shoup described in [26] an efficient method for the creation of threshold signatures in heuristic threshold schemes. A proof of security was done with the random oracle model [5].

As an example, the threshold scheme from Tal Rabin is explained briefly in section 6.1.

6.1 RSA Threshold Cryptosystem from Tal Rabin

This threshold cryptosystem was published in [22] and is an efficient, robust and optimal $(t, n)$ scheme. Also a proof of security exists. It basically is an additive $(n, n)$ threshold scheme that was modified with a second secret-sharing protocol layer to a $(t, n)$ scheme. The robustness is achieved via the verifiable secret sharing protocol from Feldman [15]. Additive means that the private key $d$ is the sum of the shares that are distributed over $n$ parties. i.e. $d = \sum_{i=1}^{n} d_i$. A shareholder $S_i, 1 \leq i \leq n$ creates his partial result $\sigma_i$ with the publication of $\sigma_i = m^{d_i} \mod N$.

A combiner can calculate the final signature by combining the $n$ shares:

$$\sigma = m^d \mod N = \prod_{i=1}^{n} \sigma_i \mod N = \prod_{i=1}^{n} m^{d_i} \mod N$$

(2)

Share backups are created by the shareholders. They distribute their shares according to the verifiable secret sharing protocol from Feldman [15] to the $n$ shareholders with a threshold of $t$. In case of a faulty or compromised
shareholder the additive share can be reconstructed with the $t$ share-backups.

7 Analysis of RSA Threshold Schemes

This section describes the demands of Ad-hoc networks on security mechanisms and gives an overview whether some of the already introduced protocols meet these demands. Basically, it is not desireable if shareholders need to interact to generate a signature. One of the main problems of such multiparty-protocols is the loss of connection that trigger a protocol-restart. With the degree of interactivity of a protocol, also the possibilities rise for malicious nodes to sabotage the protocol via suppression of packets or simulated connection-losses. If the robustness or the security of a protocol depend on the synchronicity of the underlying communication networks it is even more problematic for Ad-hoc networks. Usually, the nodes set its timers depending on the synchronicity-assumptions and after the expiration of the timers a node is classified as compromised. Strong synchronicity assumptions can not be met by Ad-hoc networks as it can not be assured if a network node is present in the network at a given time. This results in the unability to decide if a node does not want to answer for some reasons or if the node is not able to send his answer. Given this situation, diverse denial-of-service attacks can seriously undermine the robustness and the security of the whole system.

Another key point for the usability of security mechanisms in Ad-hoc networks is the size of its shares. The size should be independent from the number $n$ of shareholders to scale also for bigger networks. If the size of the shares is dependent on $n$, problems also arise when the network has to grow during operation. In this case the number of nodes that are allowed to create partial signatures has to be known before the operation of the system. If this number is estimated too high, memory is wasted. If the number turns out to be too low, a complete re-configuration of the system has to be done.

In the following paragraphs, existing protocols are evaluated for their usability in Ad-hoc networks: [16] is a $(n, n)$ threshold scheme and no proof of security was made for [13]. The criterium of optimality was not met by [18]. The size of the shares in this protocol is $10^{\log(N)}$, for a system with $t = 4$ and $n = 10$ this results in a share size of 125 Kbyte for a key size of 1024 bit. The size of the shares is dependent on the number of expected cycles of the proactivation protocol and depends therefore on the lifetime of the system. All this results in a bad efficiency for a larger number of participants. Additionally, the security of the system depends on the synchronicity of the network.

The share size of [23] is $n\log(N)$ and is linear dependent on the number of shareholders. The advantage of this protocol is its non-interactivity.

Compared to that, the protocol in [19] has an advantage in its share-size of $2\log(nN)$. It is a hybrid protocol, using a heuristic polynomial approach to transform its shares into an additive $(t, t)$ scheme before the share process happens. This was used for the proof of security of the protocol, however, was not advantageous in terms of complexity. The additive scheme was used to make the scheme proactive. After the share refresh, the additive shares are again transformed into "polynomial" ones. The protocol is robust, but only if the synchronicity-assumption is valid in the network.

One of the newer results in threshold cryptography is the RSA threshold cryptosystem from Tal Rabin [22]. The system presented in this publication gives a good and understandable overview of how a additive, proactive, robust and optimal RSA threshold scheme works. However, its usability in Ad-hoc networks is very limited. The scheme has a share size of $2\log(nN)$. As additional share backups have to be stored in each node, the memory usage for the shares is $2n\log(nN)$, linear dependent on the number of shareholders. Compared with other threshold schemes, the scheme does also use a slightly different definition of the adversaries. It makes no difference between compromised shareholders and shareholders that are not available for other reasons. This implies that the system needs the availability of $n - (t - 1)$ flawlessly working parties. This demand is hard to meet in Ad-hoc networks - demands from other threshold schemes that only need less then $t$ compromised nodes are more realistic for these network architectures.

When used in Ad-hoc networks, problems also arise due to the synchronicity assumption. The scheme requires the reconstruction of missing shares - this results in the possibility for an adversary to remove flawlessly working nodes from the network. Sabotage of the transmission channel or a simple denial of service attack against a node will result in the public reconstruction of the nodes share and the node status "compromised". In [9] the scheme of Rabin was modified to work in asynchronous environments. A prototype of a certification authority was described in this paper by Zhou et al. In [22] possibilities are mentioned to avoid the public reconstruction of shares via an additional layer of additive shares. The disadvantage of this approach is an increase of memory consumption by factor 2.

For utilization in Ad-hoc networks the protocol from Shoup [26] seems to be the most promising one so far. It is completely non-interactive and is also suitable for asynchronous networks. As disadvantage can be seen that a proof of security is only available in the random oracle model. The share size is constant, including the witnesses it is $2\log(N)$. It is possible to add new shareholders to the running system. In his publication Shoup did not make a proposal for the proactivation of his protocol. Theoretically, this could be achieved using the technique from [21]: Each Shareholder $S_i$ chooses a polynom $\delta_i(x)$ with degree $t - 1$ and the random coefficient $\delta_{i,l}(x) \in R$, $Z_M, l \in \{1, \ldots, t - 1\}$. 
This leads to the following polynomial:
\[ \delta_i(x) = \delta_{i,t-1}x^{t-1} + \delta_{i,t-2}x^{t-2} + \ldots + \delta_{i,1}x \mod M \]
(3)
\[ \delta_{i,0} = 0. \] Important here is the fact that the addition of such a polynomial to the original polynomial of the combiner \( f(x) \) leaves the secret \( d \) unchanged. This is a result from
\[ f_{new}(0) = f(0) + \delta_i(0) = d + 0 = d. \] The homomorphism \( f_{new}(x) = f(x) + \delta_i(x) = (f + \delta_i)(x) \) is responsible for the correct refresh of an old share.
Each Shareholder \( S_i \) now confidentially receives a new sub-share with the value \( \delta_i(j) \). After all shareholders have exchanged their subshares, the new share \( s_j^{new} \) is calculated as follows:
\[ s_j^{new} = s_j + \sum_{i=1}^{n} \delta_i, (j) \]
(4)
This mechanism has still to be secured against compromised shareholders. This can be done e.g. with the verifiable secret sharing mechanism from Feldmann [15]. This mechanism can also be used to change the threshold in an operating system \((t \rightarrow t')\). To achieve this goal, the nodes have to agree on polynomials \( \delta_i(x) \) with degree \( t'-1 \). Using byzantine agreement protocols, a cryptosystem can make these decisions also when compromised shareholders are present.

8 Proactivation Protocols in Ad-hoc Networks

Proactivation Protocols are highly interactive. Each node has to communicate with each other node to assign a new sub-share and the necessary witnesses to it. Given the dynamic nature of Ad-hoc networks, one can not assume the availability of all \( n \) nodes for performing the share refresh. This leads to the demand for a mechanism that does a refresh step-by-step. This results in the situation that there are no strictly defined time periods during that new shares can be assumed. Even worse, malicious nodes can increase the "window of vulnerability" if they hold back messages or simulate connection loss. It is very hard, if not impossible, to find the guilty nodes. How can the status of a node be detected if it doesn’t participate in the proactivation protocol? Is the node out of range? Is the node damaged or should it be considered malicious?
To complete the protocol, the nodes have to expire timers and exclude the nodes that do not answer from the system or mark them as faulty. The system does damage itself. This is the reason why all proactivation protocols have synchrony demands on the network. Another problem in this context is that there is no global clock, but it is necessary for the nodes to come to a decision about the right time for a share-refreshing. Byzantine agreement protocols can be very useful to solve this problem.

9 Byzantine Agreement Protocols

Byzantine Agreement Protocols can be used to make correct distributed decisions in the presence of faulty or malicious parties. The adversaries can behave arbitrary without being able to hinder the correct parties from making their decision. Up to now, byzantine agreement protocols are widely used in fault-tolerant, distributed systems. It is possible to give an upper bound for the number of faulty nodes that the system can still tolerate and make the correct decisions. For a system with \( n \) participants, this upper bound is \( f = \left\lfloor \frac{n-1}{3} \right\rfloor \). A lot of publications about byzantine agreement protocols are available, but not all of them meet the demands for the utilization in Ad-hoc networks. Some of them are not efficient enough, others assume synchrony, have very long initialization-phases or are not optimal10. One publication of such a protocol that is also suitable for Ad-hoc networks is e.g. [10]. An even more efficient version that does not work with digital signatures but does use message authentication codes instead is described in [9].

The proof of correctness of this protocol was published separately in [8]. Byzantine Agreement Protocols are highly interactive. They transmit a high number of messages to reach the same state and make the right decision. In Ad-hoc networks, such protocols can be used to enable "democratic" decisions. This is especially useful if nodes show an benign behaviour, but do not always have the same opinion about its decisions. Byzantine agreement protocols ensure that a large number of nodes was behind the decision. To work smoothly, the parameters \( f \) and \( t \) have to be set reasonably (e.g. \( t = n - f \)). It is important for the usage in Ad-hoc networks to set the parameter \( t \) not too high. The real degree of availability of the nodes has to be kept in mind. One security problem of these protocols in Ad-hoc networks is obvious: the high interactivity makes the system vulnerable to various kinds of denial-of-service attacks.

10 Conclusion

This paper gives an overview about security in Ad-hoc networks. Different algorithms are introduced and its suitability for Ad-hoc networks is discussed. Given the different constraints in such networks, also performance issues play an important role for security architectures. Future work will include the development of security architectures for securing Ad-hoc networks utilized in different scenarios and also performance studies on these architectures.

References


10 i.e. they can not tolerate the maximum number \( f \) of faulty participants.


