Technical report

Mobile Phones as Secure Gateways for Message-Based Ubiquitous Communication (Revised)

Walter Bamberger[1], Oliver Welter[1], Stephan Spitz[2], and Michael Marhöfer[3]

[1] Technische Universität München, Germany, www.ldv.ei.tum.de
[2] Giesecke & Devrient GmbH, Germany, www.gi-de.com
[3] Nokia Siemens Networks GmbH & Co. KG, Germany, www.nsn.com

August 28, 2007, final

Abstract. For ubiquitous communication self-organising ad-hoc networks become more and more important. We consider mobile phones as appropriate secure gateways to provide access to the Internet for external machines with low communication needs. A message-based approach is best in such a scenario with moving mobile phones and machines. In this paper we propose a security model for access control to the communication infrastructure, which is also message oriented. To meet the requirements of ubiquitously communicating machines, all algorithms on the sender's side are based on symmetric cryptography resulting in low computation requirements. Our sophisticated symmetric key infrastructure for access control is based on unique combinations of keys and is completed with an effective key management. This results in a carrier grade security level although many parties share the same keys. Adopting the Subscriber Identity Module as a secure storage and computing module achieves the trustworthiness of the mobile phone. This makes it possible to use the mobile phone not only as a user terminal but also as a trusted infrastructure component of the mobile network.

This document is an update of earlier work [BWS07] presented at the Workshop in Information Security Theory and Practices 2007 in Crete, Greece.

Key words: Machine-to-machine communication, message-based communication, SIM, symmetric key infrastructure, shared secrets, message authentication codes

Contents

1 Introduction

2G/3G mobile networks with packet transport capabilities are widely spread today. Besides human communication they are also used for machine-to-machine communication. This paper introduces a security architecture for a communication technology, in which the external (sending) machine is equipped with a personal area radio (PAN, like ZigBee or Bluetooth) instead of a wide area radio (WAN, like GPRS or UMTS). This keeps the module complexity on the sender's side as well as the resource allocation in the mobile network

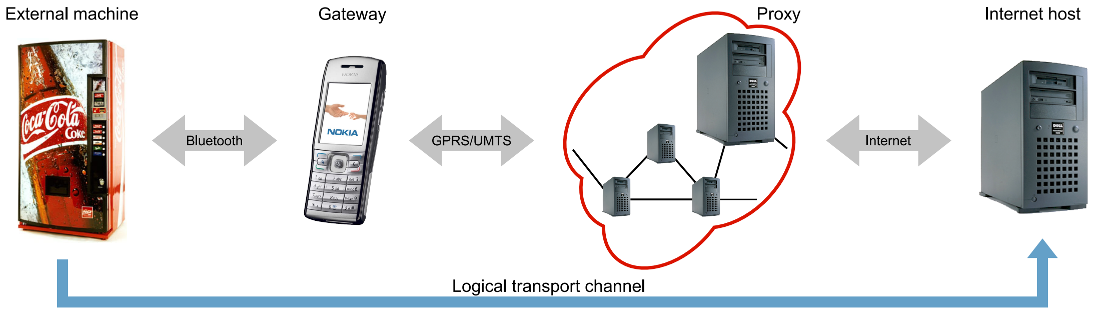

Figure 1: The considered communication scenario: An external machine should be able to send messages supported by a trusted mobile phone.

very low. Interesting applications include all sorts of vending machines, escalators, environmental sensors and many others.

Figure 1 shows the communication architecture, which uses a multi-hop relaying approach from the external machine through the gateway and the proxy to the Internet host. The communication is message-oriented, i.e. each packet contains all the routing and security information and is relayed on its own. Section 2 details a few aspects of the communication architecture further.

This paper deals with the security concerns that come along with this new communication approach. The following paragraphs introduce the main characteristics of the security concept.

As the communication is message-oriented with one or more hops, a *message-based security concept* (Section 7) must be chosen. We show how this paradigm can be integrated in the existing security architecture of the mobile network.

A *symmetric key infrastructure* (Section 6) builds the basis for message authentication here. A public key infrastructure like X.509 is not feasible as external machines have very low computation capacities and miss some prerequisites like access to a reliable time source. The proposed system makes it possible to directly implement the software on the integrated micro-controller of the Bluetooth transceiver (like the BlueCore 4 of CSR with its 16 bit micro-controller) and on a common Subscriber Identity Module (Section 4.4). A lightweight key management accompanies the key infrastructure to make it highly dynamical (Section 9). This is important, because many parties share the symmetric keys for message authentication.

As Section 6 points out, there are 256 keys in total to authenticate messages for access control; a gateway has one out of eight different subsets. Every machine operator (with its external machines) has its own unique combination of 24 keys out of those 256 keys (and not its own unique single key as usually for authentication). Then there are about 24 million different *key combinations*. This way no machine operator has the same combination of keys as any other machine operator, but four keys in common with any passing gateway. As a result, attacking a machine operator or a gateway (and disclosing all its keys) does not lead to service interruption.

Finally the *Subscriber Identity Module* (SIM) – a key component of the mobile network security – serves as a key component in this new concept too. Because the gateway should operate as an external security wall preventing unauthorised traffic in the mobile network, the functionality of the gateway is split into a trusted and an untrusted part (Section 8.2). The SIM provides the trusted environment for storing the secret keys and for security relevant calculations. The untrusted component handles the hardware access and is executed in the main processing unit of the mobile phone.

This document is structured as follows: Chapters 2–4 give some fundamentals and additional details. While the communication security spans the chapters 5–7, chapter 8 covers the platform security. For the understanding of the key management the chapters 6, 9 and 10 should be read. Finally the chapters 11 and 12 summarise the whole concept from different points of view.

2 Short Introduction to the Communication System

Figure 1 illustrates the communication architecture considered in this paper. An *external machine* (on the left hand side) wants to send a message to a host in the Internet (e.g. running a web service). For this it looks for a randomly passing mobile phone and uses it as a relay. We call such a mobile phone the *gateway* in the following. In the mobile network there is another intermediate component named *proxy*. It performs accounting and security tasks. In this paper we only discuss the unidirectional case from the external machine to the Internet host, although a bidirectional extension can be imagined.

A usual message-oriented Internet protocol stack (e.g. IP/TCP/TLS/HTTP) sets the basis for the communication between the proxy and the Internet host. The messages are tunnelled from the external machine to the proxy. Therefore only this latter network segment is of interest in this paper. It makes the difference to existing protocols and is as such the interesting part for security investigations.

Because messages of protocols like the Hypertext Transfer Protocol (HTTP) can be rather large, the external machine splits them in *packets* suitable for the short-lived ad-hoc connections between the external machine and the gateway. Those packets contain all necessary routing and security information. Therefore we say throughout this paper that the system transports packets, not messages.

3 Related Work

There are interesting activities in the research community to enhance today's mobile networks with relaying techniques. The goals are mostly coverage extension and capacity improvements at moderate costs. Pabst et al. [PWS$^+$04] provide a good starting point. We specialise our concept on machine-to-machine communication only.

For security related concepts a look at ad-hoc networks is also interesting. There are several proposals [ZH99, KZL$^+$01] to meet the ad-hoc nature with asymmetric cryptography and secret sharing techniques. Yang et al. [YML02] introduce a very localised and self-organising approach. However they do not really meet the characteristics of our communication system. Further more we want to evaluate the chances of symmetric cryptography.

Therefore a closer look at existing symmetric key infrastructures (SKI) can help for inspiration. The classical Needham-Schroeder protocol or Kerberos, but also a newer proposal of Crispo et al. [CPT04] target at user / machine authentication though. Some investigations show that this is rather different from a symmetric key infrastructure for message authentication with its keys which are shared by many devices.

Most closely related to our architecture, protocol and applications is the work of the Delay Tolerant Networking Research Group (DTNRG) in the Internet Research Task Force (IRTF). The main protocol is the Bundle Protocol [SB06], accompanied by the Bundle Security Protocol [SFW06]. Both are still drafts. The routing protocol is much more complex than ours, targeting at more applications. However the security side still has a couple of open issues, especially the key management. With our simpler protocol we can provide a thorough and practical solution.

4 Technical Fundamentals

4.1 Notes on the Subscriber Identity Module (SIM)

This paper proposes to understand a mobile phone as a gateway into the mobile network. Therefore the mobile phone plays a strongly security related role. To implement this concept, the Subscriber Identity module takes over the security critical tasks. It verifies, authorises and forwards the data, which it has received from the external machine, into the mobile network. The communication to the SIM can be established via the classical APDU interface according to ISO 7816 or via a TCP/IP protocol stack on top of an USB connection to the SIM (ETSI TS 102 600 is expected for mid 2007).

As we show in Section 9 the SIM must receive sensible key material from a server in the mobile network. Using the latest generation of Internet-enabled SIMs (like the Giesecke & Devrient GalaxySIM) a direct transport layer security (TLS) tunnel can be established between the server and the SIM. Then the mobile phone simply acts as a router between the SIM and the server. In case of an APDU based communication all data is routed through the insecure mobile phone operating system. Then additional security mechanisms have to be applied on the application level. We detail them in Section 9.

4.2 The Packet Data Protocol Context

The Packet Data Protocol context (PDP context) [3rd06] is another concept in 2G/3G networks, which is important for charging and security purposes in this concept (Sections 7.2 and 11). A mobile phone, which wants to send packet switched data (e.g. via the General Packet Radio Service (GPRS)), must request a Packet Data Protocol context first. This context can be imagined as a virtual channel. A network protocol (e.g. IP), an interface address (e.g. an IP address) and other information is associated with this virtual channel. This also includes specific routing and charging rules. In our system the mobile phone requests a certain PDP context to deliver packets to the proxy in the mobile network. Using this PDP context the routing to the proxy is possible and the data transport is not charged to the mobile phone owner's account.

Because the PDP context is requested from an early component in the core network (the Serving GPRS Support Node (SGSN)), refusing the PDP context for a given device is an efficient way to keep unwanted traffic to the proxy (which is free of charge) out of the mobile network. As Section 7.2 shows, we use this mechanism for effective attack defence.

4.3 The Pseudo-Random Function for Packet Key Generation

This paper uses the pseudo-random function (PRF) of the draft of the Transport Layer Security (TLS) standard v1.2. For convenience this section describes it shortly:

$$\begin{aligned}&\mathrm{PRF}(\mathit{secret},\ \mathit{label},\ \mathit{seed}) =\\ &\quad \mathrm{HMAC\_hash}(\mathit{secret},\ \mathrm{A}(1) + \mathit{label} + \mathit{seed}) +\\ &\quad \mathrm{HMAC\_hash}(\mathit{secret},\ \mathrm{A}(2) + \mathit{label} + \mathit{seed}) +\\ &\quad \mathrm{HMAC\_hash}(\mathit{secret},\ \mathrm{A}(3) + \mathit{label} + \mathit{seed}) +\\ &\quad \ldots\ ,\end{aligned}$$

where hash must be substituted by a specific hash algorithm as defined in the chosen cipher suite and "+" is the concatenation operator. The function A is defined as

$$\begin{aligned}\mathrm{A}(0) &= \mathit{label} + \mathit{seed}\\ \mathrm{A}(i) &= \mathrm{HMAC\_hash}(\mathit{secret},\ \mathrm{A}(i-1))\end{aligned}$$

4.4 Selected Cryptographic Algorithms on Today's Micro-Controllers

This section demonstrates in short the performance of today's micro-controllers with respect to selected cryptographic algorithms. For this several documents in literature have been studied. Table 1 on the following page compiles the results that are interesting for this paper.

To understand the table it is important to mention that most authors did not perform complete Elliptic Curve Digital Signature Algorithm (ECDSA) operations, but the basic elliptic curve algorithms that constitute a complete ECDSA operation. Therefore Table 1 gives the elliptic curve scalar multiplication timings as well. Depending on the chosen elliptic curve attributes, the elliptic curve scalar multiplication with a fixed point dominates the timing of an ECDSA signature generation, whereas the sum of both multiplication types dominates the ECDSA signature verification operation.

		16 bit			8 bit	
Model	M16C	PMS430E337HFD		CC1010	(Unknown)	ATmega128
Reference	[HNM98]	[GBKP00]		[GPW$^+$04]	[WBP01]	[GPW$^+$04]
Clock frequency	10 MHz	3 MHz	1 MHz	15 MHz	12 MHz	8 MHz
ROM / Flash	0–96 kB	32 kB		32 kB	–	128 kB
RAM (internal & external)	2–10 kB	1 kB		≈2 kB	–	4 kB
EC scalar multiplication (random point)	0.48 s	1.3 s	3.8 s	4.58 s	8.37 s	0.81 s
EC scalar multiplication (fixed point)	0.13 s	–	–	–	1.83 s	–
SHA-1 (one block)	2 ms	–	–	–	–	–
ECDSA signature generation	0.15 s	–	–	–	–	–
ECDSA signature verification	0.63 s	–	–	–	–	–
RSA signature generation	10 s	–	–	∼106.66 s	–	10.99 s
RSA signature verification	0.4 s	–	–	>4.48 s	–	0.43 s

Table 1: A collection of timings in literature.

All in all Table 1 shows that elliptic curve operations take 1–5 seconds. This is much less than RSA signature generation but much time in our scenario. In contrast an HMAC operation takes only few milliseconds. (It basically consists of the given SHA-1 operation.) Therefore we decided to propose a system based on symmetric cryptography.

Note that the results in Table 1 cannot be compared directly as they are based on different key lengths. Therefore the context of each paper is described in the following.

In [HNM98] Hasegawa et al. implemented ECDSA and RSA on a M16C of Mitsubishi Electronic Corporation. They used a key length of 1024 bit for RSA and 160 bit for ECDSA.

Guajardo et al. [GBKP00] implemented elliptic curve algorithms on a 16 bit processor of the MSP430 family of Texas Instruments. They used an elliptic curve system over $\mathrm{GF}(p)$ with $p = 2^{128} - 2^{97} - 1$. This corresponds with a RSA key length of less than 1024 bit.

Gura et al. compared two 8 bit micro-controllers in [GPW$^+$04], an ATmega128 of Atmel and a CC1010 of Chipcon. They implemented elliptic curve operations for 160 bit, 192 bit and 224 bit fields. RSA operations are measured with a key length of 1024 bit and 2048 bit. Table 1 gives the timing values for 160 bit (EC) and 1024 bit (RSA). They did not perform a complete ECDSA operation.

Finally [WBP01] looks on smart cards. They implemented elliptic curve algorithms on a derivative of the Intel 8051 (but did not give the exact processor model.) For the elliptic curve operations they use the optimal extension field $\mathrm{GF}((2^8 - 17)^{17})$, which has an order of approximately 2^{134}.

5 Requirements for Packet Transport

Defining the requirements for the security of a system mostly means compiling the necessary security services (also called security attributes, security goals, or security objectives, see [MvOV01]). For example Zhou and Haas [ZH99] explain the security requirements for ad-hoc networks in the form they are used here. Because transmitting data (in form of packets) is the main purpose of this system, this section discusses the requirements for the network security only, but not for the platform security and the key management. Those are implicitly described in their corresponding sections (numbers 8–10).

In this scenario the data should not be transmitted through an end-to-end connection. Instead a packet should be forwarded using one or more relays to reach its final destination. Each relay must verify the packet *integrity* and whether it is allowed to use the infrastructure (*access control*). This makes some kind of *message authentication* necessary.

Because the transmission in the mobile network causes costs, the mobile network operator must ensure the *non-repudiation* of origin. As a consequence another key infrastructure is set up for non-repudiation purposes, as the requirements are very different from those for access control. Note that, using symmetric keys, the mobile network operator can only prove that the packet has not been created by a third party as it is able to create verifiable packets itself. A trust relation between the machine operator and the mobile network operator is assumed, so this will not become a problem.

Finally the *anonymity* of the mobile phone (or any kind of gateway) outside the mobile network must be ensured. In addition we increase the *availability* through redundancy: An external machine may re-transmit a packet several times depending on the booked service level. It has to use different gateways for each re-transmission for security reasons.

Our system provides *confidentiality* too, but as an optional feature. There are a few applications that do not need this service but want to avoid the extra effort.

A non-security-related requirement of this work is the intent to address machines with low computation power. This corresponds with the target applications, which are characterised by low communication needs. Because of this requirement we propose a solution based on symmetric cryptography on the machine's side.

6 The Key Infrastructure

6.1 Description of the Key Infrastructure

As mentioned in the previous section two sets of symmetric keys are used. With the *access control key set* each relay and the proxy can verify that a packet (i.e. the sender) is authorised to use this mobile network for message-based communication. The *non-repudiation key set* is necessary for accounting purposes; with its keys the mobile network operator can determine the creator of the packet uniquely. In addition they are used for content encryption.

The access control key set consists of 256 keys. Each key is valid for a chosen time period (e.g. 24 months) and is replaced by a successor e.g. every 12 months (see Section 6.3). It is identified with an identification number and a version number. Section 9 describes the key management for the access control key set further. The proxy in the mobile network has access to all 256 keys.

The access control key set is divided into 8 subsets of 32 keys each. A gateway has the keys of one subset, resulting in eight groups of gateways depending on the actual subset. This ensures that the system still runs, even if all 32 keys of one gateway are compromised. The keys are deployed onto the Subscriber Identity Module card (SIM card) – an accepted security token in the world of mobile networks – and cannot leave it. This ensures a carrier grade security level. Section 8.2 details further how this module is used as a security kernel in this architecture.

An external machine has 24 keys, 3 out of each subset. During connection establishment with the gateway a subset is negotiated. Because message authentication requires shared keys, the *uniqueness of the key combination* sets the base for the security of key infrastructure here. Every machine operator has its own combination of those 24 keys; there are about 24 million of those key combinations in total. If all the keys of one machine operator are disclosed, there is no service interruption for any other machine operator with this idea. External machines and their operating companies are considered as the major risk for the secrecy of the keys.

In contrast the keys of the non-repudiation key set are not shared between the machine operators and the gateways. Each machine operator has its own unique key. The proxy in the mobile network uses these keys to verify the sender for accounting purposes. The keys are versioned as well, but the update process is not automatic. Instead the keys are exchanged during other service tasks on-site (e.g. every 5 years), so a sufficient long overlap between two consecutive key versions is required. Using only one key per machine operator reduces the size of the key database compared to individual GSM modules in each external machine.

Note: The key infrastructure introduced above makes it possible that every external machine has at least one key (actually exactly 3) in common with any single gateway. A different, opportunistic approach would avoid subsets. Instead an external machine would have got one eighth of all keys and a gateway would have got one forth. No node would have all the keys of any other node. Then the amount of key combinations would be much higher (compare this with Section 6.2) or the keys (and thus the memory complexity) per node much smaller. However there is a small chance, that the external machine has no key in common with a specific passing gateway and therefore cannot forward its packet to that gateway. Here is a simplified probability computation for that case:

We assume that every node has got 16 keys and that this is one forth of all keys. Then there are 64 keys totally and about $5 \cdot 10^{14}$ key combinations. The probability that one node has no common key with another node then computes to

$$\frac{\binom{48}{16}}{\binom{64}{16}} = 0.46\%.$$

For a higher number of keys the probability decreases further.

However at the beginning this probability decreases starting from 75% when there are exactly four gateways in total. Therefore this concept is best for a large deployment. (It is possible to start with the infrastructure given first and to switch to this opportunistic infrastructure later.)

6.2 Considerations on the Number of Keys per Node

The key infrastructure for access control is a shared infrastructure. Contradictory to this is the requirement that there may be no single point of failure – i.e. if one node becomes compromised, the attacker may not get all keys. Thus a single node may not have got all keys, although the key infrastructure is shared.

Considering the Subscriber Identity Module as a rather secure key storage, attacks on it are hard to do and occur with a low probability. Therefore we decided to divide the whole key set into only eight subsets; a gateway has the keys of one subset. If a SIM has been attacked successfully, the system still runs with $\frac{7}{8} = 87.5\%$ of its gateways (resulting in a slightly longer transmission delay).

For the external machines the situation is quite different. The machine operators are responsible for the key roll-out and this process heavily depends on the internal organisational structure of the company. Usually those companies have not much experience in secure key management. Therefore we consider the machine operators and the external machines as the weakest point for the secrecy of the keys.

This raises the following question: *How many keys should be on each external machine, so that the loss in case of a successful attack is as low as possible.* The optimisation must include the following requirements (in the given order):

1. The chance that the communication breaks for another company must be as low as possible. This means that all other company must still be able to use all passing gateways.

2. If a second external machine (of a different machine operator) is compromised, parts (at least 75%) of the system should still run, even in the worst case.

3. The memory complexity (the number of keys) in the external machine as low as possible, but in a sensible relation to the number of keys in the gateway.

The first requirement means that the machines of all other companies must have at least one key per subset different from that ones of the compromised machine. Thus there must be as many different key combinations as possible. If n denotes the number of keys per subset and k the number of keys per machine, then the number of possible key combinations

$$n_s = \binom{n}{k}.$$

This function with respect to k has its maximum at $k = n/2$. (As we discuss at the end of this section we do not use this optimum because of the second and third requirement.) Increasing and decreasing k decreases the total number of key combinations.

Keys in total	Subsets	Keys per gateway	Keys per machine operator	Key combinations
384	8	48	48	12,271,512
384	8	48	40	1,712,304
192	4	48	20	1,712,304
320	8	40	40	658,008
160	4	40	20	658,008
256	8	32	32	35,960

Table 2: For this key configurations at least one key in every subset must be different.

Keys in total	Subsets	Keys per gateway	Keys per machine operator	Key combinations
256	8	32	32	1,293,121,600
256	8	32	24	24,601,600
128	4	32	12	24,601,600

Table 3: For this key configurations at least one key in one half of the subsets must be different.

The second requirement above is a worst case requirement. It means that at most a quarter of the keys can be disclosed by attacking two machine operators at ones. Thus each external machine may have at most $256/(2 \cdot 4) = 32$ keys according to the second requirement – 4 out of each subset. This results in only 35,960 key combinations (see greyed line in Table 2). All in all, this requirement demands that k is as low as possible and is therefore contradictory to the first requirement.

Because of the small number of key combinations computed in the previous paragraph, we decided that it is sufficient, that the key combinations must be different for half of the gateways only. This means, if all keys of one machine operator are disclosed, some other machine operators (not all) can only use half of the gateways. But there is no complete service interruption. Exchanging the compromised keys on-site step-by-step (e.g. at regular service dates) can recover the full service. Table 3 gives reasonable key configurations for scenarios with this behaviour. If such behaviour is not acceptable, the key configurations of Table 2 must be chosen. Then any external machine can still use any other gateway, if all keys of one other machine operator are compromised.

Up to now our requirements lead to 32 keys in the external machine with the system behaviour of Table 3. This results in over 1 billion different key combinations – much too high for currently expected system sizes. Therefore we reduced the number of keys in the external machine to 24 (3 out of each subset) meeting the third requirement.

When choosing n as the number of keys per subset respectively gateway ($n = 32$) and k as the number of keys out of each subset per external machine ($k = 3$), the number of possible key combinations for external machines is

$$\binom{n}{k}^2 = \binom{32}{3}^2 = 24,601,600.$$

It corresponds with the number of machine operators and should be sufficiently high even for large mobile network operators. Therefore no machine operator has the same key combination like any other machine operator. Thus attacking one machine operator does not affect any other machine operator seriously.

Looking at the memory footprint the external machine must store 24 records consisting of the key identifier (8 bit), the key version (8 bit) and one key (e.g. 512 bit for SHA-1). Therefore one record has got a size of 66 Byte and the complete key table a size of 1,584 Byte. All in all, the memory usage of this security solution is in the order of an asymmetric solution, but the computation complexity is much smaller.

Note: The protocols of this communication system do not expect a certain number of keys. The number of keys can even increase or decrease and therefore adapt to the system size during run time. Only the software author must give attention, that its software modules can handle varying numbers of keys. Tables 2 and 3 compile a couple of reasonable key configurations. A mobile network operator should evaluate its needs and determine an appropriate key infrastructure, depending on the usage and threat scenario.

The bigger a subset the significantly more key combinations (machine operators) are possible. And the more subsets the smaller is the effect of disclosing all keys of a Subscriber Identity Module.

For security the interesting parameters are the number of subsets and the number of key combinations; the security of the system increases with them. Therefore the blue rows in the Table 2 and 3 fit here best. In contrast to that, the numbers of keys per gateway and especially per external machine (machine operator) determine the implementation effort.

6.3 Considerations on the Key Lifetime

When adjusting the key infrastructure another question has to be answered: How long does it take to spread out a new key?

When looking at the applications, we expect that it takes at most one year to distribute a new key to every affected external machine. Therefore the whole key set can be renewed every year; having 256 keys a new key is given out every 1.4 days.

The distribution duration of a key equals the validity overlap of two consecutive key versions. The validity time of one

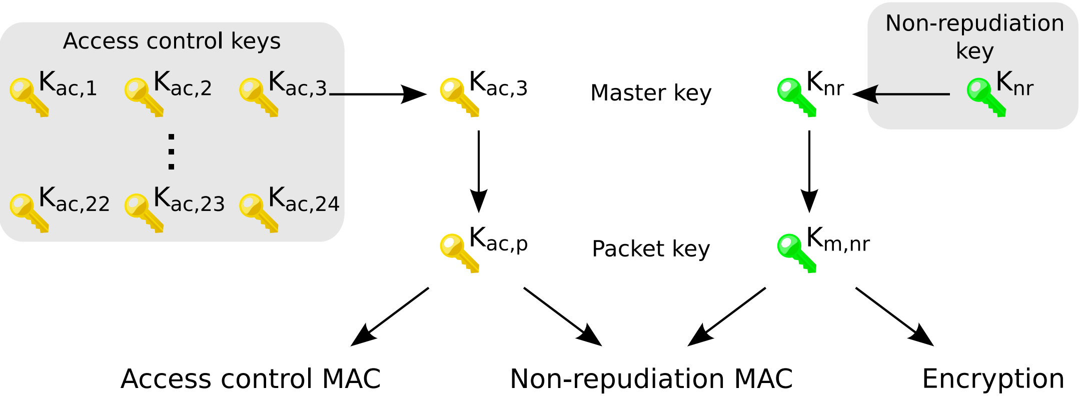

Figure 2: The key infrastructure and key processing in the external machine

key is 2 years then – one year to distribute this key and one year to distribute its successor.

Note: The external machine is responsible to keep its key set up-to-date. Section 9.2 lists all reasonable strategies.

If the expected distribution duration differs too much among the external machine types, it could be necessary to have two types of keys: keys with a high renewal interval and keys with a low renewal interval. Each subset has got keys of both types then, but an external machine has got keys of only one type.

7 Securing the Packet Transmission

With the above key infrastructure we can describe the transmission process of a packet in the following.

7.1 From the External Machine to the Gateway

First the external machine needs an *access control packet key* derived from a key out of the access control key set and a *non-repudiation packet key* derived from the machine's non-repudiation key (see Figure 2). These packet keys are recomputed for each packet and help in combination with a nonce to hinder attacks based on a large collection of data or on packets with the same payload but different keys. Because there is no end-to-end connection the packet key must be generated with a pseudo-random function and with parameters only depending on header respectively packet information (see Section 7.3). One parameter is the secret key the packet key is derived from; it is called *master key* in the following. Another parameter is the nonce which the gateway generates to prevent replay attacks. Therefore the packet keys must be computed after the external machine has connected to the gateway.

With the non-repudiation packet key the external machine encrypts the payload first. The encryption is indicated through a certain value in the content type header, as it is optional – meeting the needs of a few applications. Then two message authentication codes (MACs) must be computed, the *access control MAC* for the relaying and the *non-repudiation MAC* for accounting (see Figure 3 on the next page). To avoid the necessity of performing the hashing over the payload twice, a modification of the HMAC algorithm (a MAC based on keyed-hashing, [KBC97]) is introduced in Section 7.3. With this the non-repudiation MAC is based on both packet keys, while the access control MAC is a common HMAC over the whole packet, including the non-repudiation MAC. This makes it possible that every gateway can test the integrity of the packet and verify that the packet is authorised for this service; it uses a standard algorithm (HMAC) for this. In addition the proxy can prove that the sender address indicates the right customer.

All in all (Figure 2) when the external machine has found a gateway, it receives the number of the access control key set and a nonce, chooses an appropriate master key out of that key set, optionally encrypts the payload, computes both MACs and finally delivers the packet to the gateway.

When the connection has been closed, the gateway validates the packet. This can be implemented on the Subscriber Identity Module. Section 8.2 gives more security related details on this process.

7.2 From the Gateway to the Proxy

If the packet is valid and as soon as there is mobile coverage, the gateway (in form of a mobile phone) sends the packet to the proxy in the mobile network. First a software component in the untrusted area of the mobile phone requests a specific Packet Data Protocol context (PDP context) from the Serving GPRS Support Node. With this PDP context the mobile phone can access the proxy. It delivers the packet via an unsecured Hypertext Transfer Protocol (HTTP) connection. Because a packet is usually much smaller than 100 kB, the use of an authentication protocol like the Transport Layer Protocol (TLS, [DR06]) would lead to a high overhead. It is more efficient to accept every packet (unauthorised), limit the packet size and verify both MACs. Therefore it is better in this situation to react effectively on attacks, instead of preventing them with a high effort – although this channel into the mobile network is very vulnerable, because it is not charged to the mobile phone owner's account.

If one of the MACs or the combination of the non-repudiation key and the access control key is not valid (one non-repudiation key and exactly 24 access control keys are assigned to a machine operator), the proxy can detect an attack. The nonce and the various numbers in the header (see Figure 3 on the next page) make it possible to detect replay attacks. In addition optional destination filters at the proxy can protect companies if their non-repudiation key has been compromised. There are several measures available to react

<table>
<tr><td colspan="3">. . .</td></tr>
<tr><td>Message number</td><td>Sequence number</td><td>Re-transmission number</td></tr>
<tr><td colspan="3">. . .</td></tr>
</table>

<table>
<tr><td>Content type</td><td>Header length</td><td>Authentication parameters</td></tr>
<tr><td colspan="3">Nonce</td></tr>
<tr><td colspan="3">Access control MAC</td></tr>
<tr><td colspan="3">Non-repudiation MAC</td></tr>
</table>

Figure 3: Header of each packet

on those attacks:

- Traffic from a manipulated gateway can be suppressed refusing the Packet Data Protocol context for them. The gateway cannot send any data without charging anymore.
- Further criminal acts can lead to legal consequences, because the mobile phone owner is known.
- Attacks from devices behind the gateway are detected by the gateway. Only newly compromised keys could pass the gateway.
- Key management mechanisms as described in Section 9 make it possible to react precociously on compromised keys.

All in all, we have seen that the use of the Subscriber Identity Module as a secure kernel combined with other existing security mechanisms of the mobile network makes it possible to keep unwanted traffic out of the mobile network. This architecture extends the 3G network efficiently for message-based external access.

7.3 Implementation Details

Packet Key Computation

A limited number of 256 keys is used among many devices and many messages respectively packets. This makes it necessary to use a packet key (k^p) for the MAC computation instead of one of those 24 master keys ($k_{ac,i}$, $i \in \{1, \ldots, 256\}$) directly (see Figure 2 on the preceding page). The algorithm to derive a packet key from a master key is the same for both, the access control packet key and the non-repudiation packet key. The only difference is the chosen master key.

The packet key must be derived with a pseudo-random function (PRF) from a master key. In addition a few header fields and a nonce randomise the key generation. The nonce is the one that the gateway generates and that Figure 3 shows. The function must be able to provide a bit stream with variable length depending on the actual hash function in the HMAC computation.

The pseudo-random function as defined in the draft of the Transport Layer Security protocol (TLS) v1.2 [DR06] is chosen here. It has the form $\mathrm{PRF}(secret, label, seed)$. (For convenience Section 4.3 gives it in short.) The *secret* is the master key. The *label* can be "access control key" or "non-repudiation key" depending on the master key. The nonce concatenated with the source address, the destination address, the message number and the sequence number builds up the *seed*. All input values of the PRF are part of the header (see Figure 3). Therefore all hops equipped with the master key can and must verify the access control MAC before forwarding the packet.

Note: The nonce here may not be used twice, but it may be counted sequentially; the (pseudo-) randomness is provided by the PRF.

Message Authentication Code Computation

As Section 5 explains, two MACs are necessary for two different purposes: one to control the access to the relaying mechanism and another one to prove the origin of the packet for accounting purposes. Using the conventional HMAC algorithm, this would result in two hash computations over the complete packet. Since this system targets at external machines with low computation power, a modified combined method is proposed in the following. As a result the access control MAC can be verified with the usual HMAC verification algorithm, whereas the verification of the non-repudiation MAC needs both keys – the access control packet key and the non-repudiation packet key.

For the MAC generation an HMAC operation over the packet p (without the not yet computed MACs) is performed with the access control packet key k^p_{ac} first.

$$h_i = \mathrm{HMAC}(k^p_{ac}, p) \tag{1}$$

The non-repudiation MAC h_{nr} can be derived from this intermediate result with the non-repudiation packet key k^p_{nr}:

$$h_{nr} = \mathrm{HMAC}(k^p_{nr}, h_i + nonce)$$

To verify this MAC both keys (k^p_{ac} and k^p_{nr}) must be known. This is true for the external machine and the proxy.

To complete the access control MAC, h_{nr} must be appended to the HMAC operation of (1). The state of that first HMAC computation must be preserved until this last HMAC computation. Then it is possible to verify the MAC with the usual HMAC algorithm over the complete packet including the non-repudiation MAC, but in a slightly different order.

Both MACs can be inserted in the packet as shown in Figure 3 on the previous page.

8 Platform Security

8.1 Key Protection in the External Machine

One design goal for this communication system has been to simplify and cheapen the device and the key management. A machine operator should easily be able to add a machine or remove one. Registering the machine via a web page to receive a machine address should be all. In addition it should be possible to make small and cheap external machines.

When abandoning the Subscriber Identity Module, the security architecture, processes and implementation turn into company-internal matters. Then the key deployment becomes the main weakness for the key protection (see Section 10). Further an attacker could try to break in an external machine or to attack the Bluetooth interface to disclose the keys. The latter one is nearly impossible as long as the implementation does not offer server services. To ensure a minimal security level, we recommend that the mobile network operator compiles a security policy.

To control physical access to the key storage, all keys should be saved in an encrypted form. The key for this operation can be stored in the machine's read-only memory (ROM). Accessing the ROM must be as hard as possible for an attacker. This concept still leaves attacks feasibly, but hinders them.

To increase the security the machine operator can use smart cards for key storage and processing. The key derivation algorithm could be implemented on it. However timing and cost concerns limit this solution.

All in all, it is important to convince the machine operator of his own interest in the key safeness. In case all its keys become disclosed, the communication facility of all its machines breaks.

8.2 Processing within the Gateway

In this concept the new extension to the security of the 3G network is the understanding of a mobile phone as a trusted gateway for message-based access. The trust originates from two measures: First we use the Subscriber Identity Module as a secure key storage and trusted processing platform, second the mobile network operator can associate each packet with a mobile phone and thus with a real world person. Therefore the gateway is considered to be a mobile phone with a SIM throughout this section.

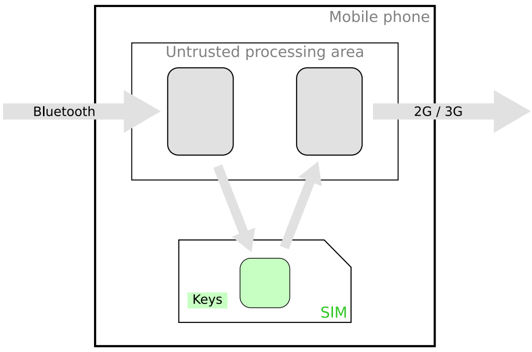

Figure 4: Security architecture of the gateway

Figure 4 shows that a server module in the main processing area of the mobile phone accepts an incoming packet from an external machine. The symmetric keys for the access control MAC verification must be stored in a trusted environment. Therefore the server module forwards the packet to the SIM card next. (Section 4.1 describes the different types of data exchange with the SIM.) A small software module in the SIM verifies the access control MAC (an HMAC). If it is valid, the SIM sends the packet back to the main processor or directly to the proxy (depending on the capabilities of the SIM). Otherwise it simply drops the packet without further notifications backwards. This ensures that faked packets do not pass the mobile phone. The only chance for an attacker to send packets through the gateway consists in revealing a valid access control key. The disclosure of the key will be detected at the proxy, because of a wrong non-repudiation MAC. The key management system (see Section 9) provides methods for key revocation, so once the attack is noticed, the abuse of the network is intercepted. All sensitive data is handled inside the trusted environment of the SIM and no secrets are visible from the untrusted domain at any time.

Note: The Subscriber Identity Module is a widespread secure platform. Today it already has standardised capabilities for additional services. The development is still ongoing mostly driven by the 3rd Generation Partnership Program (3GPP) and the GlobalPlatform. Therefore we expect improved hardware capabilities (e.g. more memory, faster and more powerful interfaces) and software capabilities (e.g. simpler application management).

Nonetheless the introduced architecture can be realised in the same way with a Mobile Trusted Module (MTM) as specified by the Trusted Computing Group (TCG). A trusted processing and storage area based on a Mobile Trusted Module would execute the same tasks and would interact with the untrusted area in a very similar way. Therefore our concept can easily be transformed into an MTM based solution.

9 Management of the Access Control Keys

The system architecture relies on the secrecy of a set of keys for access control that is shared among all participants. Therefore an appropriate key management must make the key infrastructure highly dynamic. This chapter shows the key roll-out, the key renewal and the key revocation process.

In the following the proxy under control of the mobile network provider is considered equal with the central key management server. Even if the main system uses symmetric cryptography, each Subscriber Identity Module contains an asymmetric key pair used for mutual authentication during key roll-out and key revocation.

9.1 Key Roll-Out

The SIM cards are delivered to the customers with an initial version of the secure application, an individual key pair and certificates necessary to authenticate themselves against the key management server. On first start-up the Subscriber Identity Module connects to the management system via a secure HTTP connection with mutual authentication. The asymmetric key pair is used for this. Through this secure tunnel it receives a current version of the software and the current key set.

In the external machine the initial set of keys comes with the hardware roll-out; thus the keys leave the protected environment of the network operator. This deployment is a critical but company specific task; thus it is not in the scope of this work. Sections 8.1 and 10 detail further thoughts on this topic.

9.2 Key Renewal

To allow key versioning each key index is extended by an additional version number. A new version number is the increment-by-one of its direct predecessor value. This enables the devices to decide if a presented key is newer or older than the one it currently uses without having access to the whole key history. In addition each key is associated with an expiration date.

The key renewal is done in two steps: In step I the new keys are made available on the key management server, from where the gateways can fetch them. A gateway starts the update procedure, when the expiration date has been exceeded, when the proxy rejects a delivered packet because of an outdated key, or when a packet of an external machine indicates a newer key version.

First the gateway sends a list of the key versions in its local key store to the server via an HTTP connection. The server compares the list with the key version in the repository and returns updates for all outdated keys. In this key renewal response the new key is encrypted with its predecessor, so no further authentication or transport encryption needs to be done (for details about the key renewal response see Section 9.4). The device must store the new key and the key renewal response for later use.

This procedure only works as long as the renewed key in the gateway is still valid (according to the expiration date and the key revocation list). In any other case the new key must be exchanged secured with the asymmetric key pair of the gateway. Depending on the SIM capabilities, either the SIM establishes a direct TLS tunnel to the key management server or the key management server must encrypt the new key on the application layer. (Note that this asymmetric key pair should not be used for encryption directly because it is already intended for signing. Instead an authenticated key exchange method must be applied first.)

In step II the new keys are distributed to the external machines. These are responsible to keep their keys up-to-date. There are several reasonable strategies to trigger the key renewal process:

1. If the machine maintains a clock, it can use the key expiration date as an event for key update.
2. If the machine sends packets very regularly (e.g. every week) it can use a different key for every packet and start the key update when a gateway complains an outdated key.
3. If the machine sends a packet more than once a month, it can query a key version list ((ID,version) tuples) every time and compare it with its internal list. When it finds differences the machine requests the new keys.
4. If the machine does not fit in the previous groups, it must find its own (internal) way to trigger a key update. Then it looks for a passing gateway, requests the current key version list and updates its keys finally. It does so until it has updated the keys of all subsets.

The key update procedure is basically the same as between the proxy and the gateway. It uses the same key renewal response as given in Section 9.4. Again only a difference of one in the version number can be bridged by this mechanism. If the gap is larger or the key has been revoked, a service technician must come on-site (compare Section 10). In the meanwhile the machine could use one of the remaining keys.

If the key version presented by the external machine is newer than the one in the gateway, the communication request is accepted but the packet is kept in a quarantined state. As soon as a connection to the key management server is available, the gateway performs a key update and evaluates the packet using the new keys.

Because the ad-hoc connection between the external machine and the gateway is very short-lived, some further considerations are necessary about the software architecture in the mobile phone; the access to the Subscriber Identity Module is too slow. Section 9.4 details this further.

Note: The trigger strategy three has the drawback of an additional communication on the short-lived ad-hoc segment. Some application could not accept this delay.

The strategy two is very opportunistic and can lead to higher transmission delays (because of the gateway rejecting the packet). The machine designer should weigh up which strategy is best suited for a given application.

9.3 Key Revocation

If one of the keys becomes compromised, it may not be used and accepted anymore. The key management server declares the key as revoked. It distributes key revocation notes to the gateways, and the proxy rejects all packets secured with that key. Section 9.4 gives details about the key revocation note.

There are several mechanisms to inform the gateway about revoked keys:

- When the proxy rejects a packet with an error code indicating a revoked key, the gateway connects to the key management server to update all its keys.
- Each time the gateway connects to the key management server (e.g. to regularly renew a key), it receives a list of all valid key versions and a list of all keys that have been revoked but would still be valid according to their expiration date.
- The proxy passes the key revocation note to all connecting gateways for e.g. four weeks. This strategy spreads the key revocation note very fast, but involves a certain communication overhead. Nonetheless this seems to be a good heuristic, because it informs very active gateways first.

A system implementation should use a mixture or all of these strategies. As soon as a gateway has received a key revocation note, it updates its corresponding key (as described in Section 9.2) and rejects all affected packets. The gateway keeps the key revocation note in its untrusted processing area to be able to quickly forward it to an external machine (Section 9.4).

There is no secure way to update compromised keys inside the external machine. The knowledge of the other keys is not sufficient to receive the new version of the key. Even if a key exchange is not possible, it is wise to push the key revocation note to the machine, so it no longer sends packets with an invalid key.

9.4 Implementation Details

Key Renewal Response

ID	Version	New key (encrypted)	MAC

Figure 5: The format of the key renewal response

For each outdated key the key management server sends a key renewal response to the gateway as shown in Figure 5. It contains the new key in an encrypted form using the bit-wise difference between the old and the new key ($k_i^n \oplus k_i^{n-1}$). To proof the authenticity of the response, an HMAC using the old key is appended ($\mathrm{HMAC}(k_i^{n-1}, m)$). If the HMAC is valid, the key renewal response is considered authentic and the gateway can recover the new key k_i^n with another XOR operation.

The above response is saved to be used for the key renewal between the gateway and the external machine as well.

Key Revocation Note

ID	Version	MAC

Figure 6: The format of the key revocation note

The key revocation note (Figure 6) contains the key identifier and the key version to exactly determine one single key. An HMAC secures and authorises it. The key management server uses the revoked (invalid) key to compute the HMAC. An invalid key may be used for this specific purpose, because even the attacker (who compromised this key) may generate the key revocation note – or saying it in other words: If anyone generates this note, the key has in fact been compromised.

Software Architecture in the Gateway for Key Renewal

The key renewal between a gateway and an external machine imposes some problems based on the nature of ad-hoc networks. The time slot available for communication between the Subscriber Identity Module and the machine can be very short and dispatching a packet from the Bluetooth stack to the trusted execution environment on the SIM card has a high latency. To provide a fast response on key version errors and for key renewal responses, the version list, the encrypted key material, and the key revocation notes are stored (in copy) in the untrusted area of the gateway. Then the gateway can immediately respond on packets with outdated keys and on key renewal requests. Because no unsecured confidential data is involved, the update process can be executed over any untrusted media to any kind of gateway or external machine.

10 Management of the Non-Repudiation Keys

The non-repudiation keys are known only to two parties – the machine operator and the mobile network operator. Therefore a complex key infrastructure as introduced above is not necessary here. Instead these keys are considered to be more long-lived. If we assume that a service technician comes onsite at least once in two years, the key renewal process does not lead to an additional effort.

The key deployment demands a secure process within the company of the machine operator. It depends strongly on the organisational structure there and is therefore out of the scope of this work. Some thoughts on it though include that all keys (the non-repudiation and the access control keys) reside in an encrypted form on a cheap exchangeable flash memory (something like SD cards). All machines have a super key in their fixed flash to access their keys (compare with Section 8.2). This way the keys do not leave a certain area in the company unencryptedly.

11 Discussion of Selected Attacks

This section focuses on the vulnerabilities the proposed system imposes. Attacks on Bluetooth ([Blu07, Bia05a, Bia05b] are good points of entry) or the mobile network are out of the scope because these technologies are present with or without our system and those attacks are mostly implementation dependent.

A major thread on today's communication systems are denial-of-service (DoS) attacks as they tend to be easy to execute. Looking at the external machine such an attack could be executed by faking a legal gateway and capturing all packets a machine wants to send. Two methods can be combined to prevent this. First, it is part of the communication concept, that an external machine may send a packet several times according to the booked service level. For retransmission the external machine is required to use different gateways to complicate a successful attack. Second, machine operators who need a very high security level can configure their machines to authenticate the gateway (with the access control keys). Because this costs much time, this decision should be well considered.

Next someone could try to attack the message authentication codes of a captured packet. To hinder this, packet keys have been introduced. But basically it depends on the hash function, whether such an attack is possible. The HMAC specification [KBC97] details the requirements for an appropriate hash function.

The next component is the gateway. It can detect all kinds of faked packets (including replayed packets), if it chooses the nonce appropriately. Only wrong non-repudiation MACs cannot be found. However the attacker must know correct access control keys in this case.

The application on the Subscriber Identity Module must be written with security in mind, as a successful attack on it might reveal a whole subset of access control keys and possibly one authentication key pair. In general we consider a successful attack on the card hard but possible. However the keys on the SIM are not sufficient to successfully send a packet into the Internet. A non-repudiation key is necessary too. Therefore the economic benefit in attacking a SIM is limited.

Finally the MACs, the combination of the keys, the nonce and the various numbers in the header of each packet help to detect all kinds of attacks on the proxy in the mobile network. Revoking compromised keys and refusing the PDP context for the affected gateways are effective measures in this situation (compare with Section 7.2).

Spreading a shared secret over many entities increases the probability of a compromisation. Alternatives like asymmetric cryptography have many other downsides. Therefore we designed a dynamic key infrastructure (with key renewal and revocation) based on unique key combinations; it keeps nearly unaffected if either a SIM or an external machine is compromised.

12 Conclusion

This paper introduced the security concept for a new communication concept. The communication system provides message based access into the mobile network for machine-to-machine communication. It is suitable for machines with low to very low unidirectional communication activity, and it offers low complex and cheap modules in the sending machine as well as low resource allocation in the mobile network for ubiquitous communication.

The presented message-oriented security concept uses a sophisticated symmetric key infrastructure for access control. It is based on unique combinations of keys on the sender's side. This idea makes it possible to reach a high security level and availability despite the use of widely shared keys. Together with the key management, this approach offers a carrier grade security and reliability level for ubiquitous communication.

Finally this paper extends mobile phones with a secure gateway functionality. The wide spreading of mobile phones today realises one important characteristic of ubiquitousness. In this approach the mobile phone acts as a gateway and firewall for message based access into the mobile network. The widely accepted Subscriber Identity Module builds the important security kernel for this functionality.

References

[3rd06] 3rd Generation Partnership Project. *3GPP TS 23.060 V7.3.0: General Packet Radio Service (GPRS); Service description; Stage 2*, December 2006. URL http://www.3gpp.org/ftp/Specs/html-info/23060.htm.

[Bia05a] M. Bialoglowy. Bluetooth security review, part 1. Web page, April 2005. URL http://www.securityfocus.com/infocus/1830.

[Bia05b] M. Bialoglowy. Bluetooth security review, part 2. Web page, May 2005. URL http://www.securityfocus.com/infocus/1836.

[Blu07] Security. Web page, 2007. URL http://www.bluetooth.com/Bluetooth/Learn/Security.

[BWS07] Walter Bamberger, Oliver Welter, and Stephan Spitz. Mobile phones as secure gateways for message-based ubiquitous communication. In Damien Sauveron, Konstantinos Markantonakis, Angelos Bilas, and Jean-Jacques Quisquater, editors, *Information Security Theory and Practices*, volume 4462 of *LNCS*, pages 175–188. Springer, 2007. ISBN 978-3-540-72353-0. URL http://dx.doi.org/10.1007/978-3-540-72354-7_15.

[CPT04] B. Crispo, B.C. Popescu, and A.S Tanenbaum. Symmetric key authentication services revisited. In *Information Security and Privacy*, volume 3108 of *Lecture Notes in Computer Science*, pages 248–261. Springer, Berlin / Heidelberg, 2004. ISBN 978-3-540-22379-5. ISSN 0302-9743 (Print) 1611-3349 (Online). URL http://www.cs.vu.nl/~crispo/ski.html.

[DR06] T. Dierks and E. Rescorla. *The TLS protocol. Version 1.2 (Internet draft)*. Internet Engineering Task Force, October 2006. URL http://tools.ietf.org/id/draft-ietf-tls-rfc4346-bis-02.txt.

[GBKP00] Jorge Guajardo, Rainer Blümel, Uwe Krieger, and Christof Paar. Efficient implementation of elliptic curve cryptosystems on the TI MSP430x33x family of microcontrollers. In *Public Key Cryptography: 4th International Workshop on Practice and Theory in Public Key Cryptosystems, PKC 2001, Cheju Island, Korea, February 13-15, 2001. Proceedings*, volume 1992 of *Lecture Notes in Computer Science*, pages 365–382. Springer Berlin / Heidelberg, 2000. ISSN 0302-9743 (Print) 1611-3349 (Online). URL http://www.springerlink.com/content/04n1xe9r1297ehne.

[GPW+04] Nils Gura, Arun Patel, Arvinderpal Wander, Hans Eberle, and Sheueling Chang Shantz. Comparing elliptic curve cryptography and RSA on 8-bit CPUs. In *Cryptographic Hardware and Embedded Systems – CHES 2004*, volume 3156 of *Lecture Notes in Computer Science*, pages 119–132. Springer Berlin / Heidelberg, 2004. ISBN 978-3-540-22666-6. ISSN 0302-9743 (Print) 1611-3349 (Online). URL http://www.springerlink.com/content/87aejjlhqn6fuxpy.

[HNM98] Toshio Hasegawa, Junko Nakajima, and Mitsuru Matsui. A practical implementation of elliptic curve cryptosystems over GF(p) on a 16-bit microcomputer. In *Public Key Cryptography*, volume 1431 of *Lecture Notes in Computer Science*, pages 182–194. Springer Berlin / Heidelberg, 1998. ISBN 3-540-64693-0. ISSN 0302-9743 (Print) 1611-3349 (Online). URL http://www.springerlink.com/content/ur4p561860141805.

[KBC97] H. Krawczyk, M. Bellare, and R. Canetti. *HMAC: Keyed-Hashing for Message Authentication. RFC 2104*. Internet Engineering Task Force, February 1997. URL ftp://ftp.rfc-editor.org/in-notes/rfc2104.txt.

[KZL+01] J. Kong, P. Zerfos, H. Luo, S. Lu, and L. Zhang. Providing robust and ubiquitous security support for mobile ad-hoc networks. In *Ninth International Conference on Network Protocols*, pages 251–260. IEEE Computer Society, November 2001. ISBN 0-7695-1429-4. URL http://ieeexplore.ieee.org/iel5/7792/21404/00992905.pdf.

[MvOV01] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 5th edition, August 2001. ISBN 0-8493-8523-7. URL http://www.cacr.math.uwaterloo.ca/hac/.

[PWS+04] R. Pabst, B.H. Walke, D.C. Schultz, P. Herhold, H. Yanikomeroglu, S. Mukherjee, H. Viswanathan, M. Lott, W. Zirwas, M. Dohler, H. Aghvami, D.D. Falconer, and G.P. Fettweis. Relay-based deployment concepts for wireless and mobile broadband radio. *Communications Magazine, IEEE*, 42(9):80–89, September 2004. URL http://ieeexplore.ieee.org/iel5/35/29478/01336724.pdf.

[SB06] K. Scott and S. Burleigh. *Bundle Protocol Specification (Internet draft)*. IRTF, December 2006. URL http://www.ietf.org/internet-drafts/draft-irtf-dtnrg-bundle-spec-08.txt.

[SFW06] S. Symington, S. Farrell, and H. Weiss. *Bundle Security Protocol Specification (Internet draft)*. IRTF, October 2006. URL http://www.ietf.org/internet-drafts/draft-irtf-dtnrg-bundle-security-02.txt.

[WBP01] Adam D. Woodbury, Daniel V. Bailey, and Christof Paar. Elliptic curve cryptography on smart cards without coprocessors. In *Proceedings of the fourth working conference on smart card research and advanced applications*, pages 71–92. Kluwer Academic Publishers, Norwell, MA, USA, 2001. ISBN 0-7923-7953-5. URL http://www.crypto.ruhr-uni-bochum.de/imperia/md/content/texte/woodburybaileypaarcardis.pdf.

[YML02] Hao Yang, Xiaoqiao Meng, and Songwu Lu. Self-organized network-layer security in mobile ad hoc networks. In *WiSE '02: Proceedings of the 3rd ACM workshop on Wireless security*, pages 11–20. ACM Press, New York, NY, USA, 2002. ISBN 1-58113-585-8. URL http://doi.acm.org/10.1145/570681.570683.

[ZH99] Lidong Zhou and Z.J. Haas. Securing ad hoc networks. *Network, IEEE*, 13(6):24–30, Nov/Dec 1999. URL http://ieeexplore.ieee.org/iel5/65/17493/00806983.pdf.