On the Security and Composability of the One Time Pad

Dominik Raub, Rainer Steinwandt, and Jörn Müller-Quade

IAKS, Arbeitsgruppe Systemsicherheit, Prof. Dr. Th. Beth
Fakultät für Informatik, Universität Karlsruhe (TH),
Am Fasanengarten 5, 76131 Karlsruhe, Germany
{draub, steinwan, muellerq}@ira.uka.de

Abstract. Motivated by a potentially flawed deployment of the one time pad in a recent quantum cryptographic application securing a bank transfer [12], we show how to implement a statistically secure system for message passing, that is, a channel with negligible failure rate secure against unbounded adversaries, using a *one time pad* based cryptosystem. We prove the security of our system in the framework put forward by Backes, Pfitzmann, and Waidner [11, 2, 3].

1 Introduction

It is well known that the *one time pad* (OTP) is perfectly concealing, i.e. that given an arbitrary ciphertext $c \in \{0,1\}^n$, the probability of any message $m \in M \subseteq \{0,1\}^n$ is $P(m|c) = P(m)$ where M denotes the message space. Therefore one time pad based encryption is the obvious choice when dealing with unbounded adversaries. However, the one time pad on its own does not suffice to implement secure message passing, as it is "malleable" in the sense that plaintext bits can be flipped by flipping the corresponding ciphertext bit.

Recently, a bank transfer of EUR 3000 was secured by quantum cryptography [12], i. e., a quantum key agreement scheme was used to establish a shared secret and a one time pad encrypted money transfer form was sent. However, in the experiment the integrity of the message was not secured which can have devastating consequences (cf. [6, Section 1.4]): Say, the bank transfer form itself contains no authentication mechanism and there is a known position where the amount of money is specified in digits. Then an adversary can flip bits at these positions. Such a change cannot be noticed at the bank and the resulting cleartext would look like the original message, but showing a different amount of money. Hence the security of a bank transfer as described in [12] cannot be concluded from the security of the (authenticated!) quantum key agreement protocol alone.

Therefore, to implement secure message passing, the one time pad needs to be combined with some kind of authentication scheme, to ensure non-malleability and of course authenticity. Note that this already implies, that we cannot implement perfectly secure message passing using the one time pad. In fact there is no way to classically implement perfectly secure message passing over unauthenticated channels, because the adversary always has a negligible chance to guess the secret we use for authentication correctly and thus introduce a forged message. So statistical security is the best we can do. This limitation also applies to quantum cryptographic schemes like [5] that use a statistically secure key exchange on quantum basis but classical schemes for encryption and authentication. In this work we will give a secure message passing protocol that achieves statistical security and prove its security against unbounded adversaries in the formal framework developed by Backes, Pfitzmann, and Waidner [11, 2, 3]. For being able to deal with an unbounded adversary, we will use an authentication scheme described by Stinson [14, Chapter 10.3], but note that any other statistically secure and composable authentication scheme serves our purpose as well. Secure message passing in the presence of a computationally bounded adversary is treated by Canetti and Krawczyk in [7].

There are a number of other issues with the one time pad. Since the OTP is a stream cipher, synchronization needs to be maintained. If the adversary suppresses a message, the subsequent messages should still be readable. Therefore the current position in the key used for encryption needs to be transmitted with the ciphertext. The key position must remain readable, so it may not be encrypted with the message. It should also be authenticated with the message, as otherwise the adversary may modify the key position. Then the message would decrypt to random bits on the receiver side, yet appear authenticated, so the receiver would be under the impression that the sender produces nonsensical messages.

It is usually advantageous to first encrypt a message and then authenticate the ciphertext, as it is desirable to detect forged messages before decryption to reduce workload for the receiver and lessen the susceptibility to denial of service attacks. In fact, first encrypting and then authenticating the ciphertext is the only generically secure procedure for implementing secure channels [9]. That is, a protocol intended to implement a secure channel based on an arbitrary symmetric encryption primitive (secure against chosen plaintext attack) and an arbitrary message authentication function (secure against chosen message attacks) will in general only be secure if it encrypts the message and then authenticates the ciphertext. Since we wish to generalize our results from the one time pad and our specific authentication function to other ciphers and authentication functions, we will therefore use the generically secure encrypt-then-authenticate scheme.

Heeding the arguments and restrictions above, the basic structure of our real system is nearly determined. Since we discuss the one time pad (and for reasons of applicability), we will choose the alphabet $\Sigma = \mathbb{F}_2$ for all subsequent discussions.

2 The Ideal Model

In a first step we need to define precisely, what secure message passing is supposed to mean. To this end we specify an ideal functionality TH, that obviously ensures the secrecy and authenticity of our messages, passes all information leaked through necessary imperfections of the message passing system to the adversary and offers the adversary a well defined interface to perform all actions the message passing system cannot prevent (i.e. reordering or suppressing messages). So it becomes obvious what the system does *not* conceal and *not* prevent.

In defining the ideal model for secure message transmission, we largely follow [11]. Since the handling of multiple sessions can be derived from the handling of individual sessions by means of the composition theorem, we can restrict ourselves to considering only a single session as is done in [7]. We do not need session identifiers, since multiple sessions can be distinguished by different port names. Also, differing from [11], we will for now only investigate unidirectional message transmission, with one designated sender and one designated receiver. This is no restriction, since arbitrary networks can be composed from secure unidirectional point to point connections, and security of these networks follows by composability [11] (this is analogous to [7]). There is a further reason, why it is sensible to first restrict oneself to a unidirectional connection:

Caveat 1 *As we investigate stream ciphers (more specifically the one time pad) in an asynchronous framework, any two participants need a* separate key *for* each unidirectional connection.

Rational. If two parties were to use the same key string for a bidirectional connection it would be hard to ensure, that a specific subsequence of the key is not used twice (e.g. simultaneously by each partner sending out some message). But this would undermine the security of the protocol, since by calculating the sum of two such ciphertexts the key can be cancelled. The resulting sum of two plaintexts may be deciphered using statistical methods. □

Now let $s \in \mathbb{N}_{>0}$ and L be a non-zero polynomial with coefficients in $\mathbb{N}$, where s denotes the maximum number of messages the sender may send, and L the maximum message length as a function of the security parameter k.

All machines given throughout this work will be initialized to a state corresponding to the security parameter k in accordance with the [11] framework and the adversary is always the master scheduler, all buffers not explicitly scheduled by another machine are scheduled by the adversary.

Since we only admit two participants, the sender and the receiver, a single dishonest party can, due to the nature of the message transmission task, already disclose all relevant information to the adversary A. Hence we require nothing in case one (or both) parties are corrupted and turn control over to the adversary—all messages are passed directly to the adversary who may also send arbitrary messages to the honest users ("environment") H. Therefore we only need to discuss the case where the set of honest participants $\mathcal{H}$ is the set of all participants $\mathcal{M}$, i.e. $\mathcal{H} = \mathcal{M} = \{1, 2\}$, where 1 is the sender and 2 the receiver. We now define the ideal system for secure message transmission as

$$Sys_{s,L}^{\mathsf{secmsg,ideal}} = \{(\{\mathsf{TH}\}, S)\} \tag{1}$$

where $\mathsf{ports}(\mathsf{TH}) = \{\mathsf{in}_u?, \mathsf{out}_u!, \mathsf{out}_u^{\triangleleft}! | u \in \mathcal{H}\} \cup \{\mathsf{in}_{\mathsf{sim}}?, \mathsf{out}_{\mathsf{sim}}!, \mathsf{out}_{\mathsf{sim}}{}^{\triangleleft}!\}$ and the specified ports are given by $S^c := \{\mathsf{in}_u!, \mathsf{out}_u? | u \in \mathcal{H}\}$.

TH maintains data structures $init_1, init_2, key_1, key_2 \in \{0, 1\}$, $sc \in \{0, \ldots, s\}$ initialized to 0, and a list *deliver* initially empty. $init_u$ stores if user u has initiated key exchange, key_u stores if user u would have received his set of keys (both encryption and authentication keys) in the real model, *deliver* holds the messages due for delivery until the adversary schedules them. The state-transition function of TH is given by the rules below. Inputs not treated explicitly are ignored. If an input triggers a non-empty output, we say the machine accepts it.

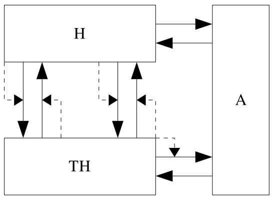

Figure 1: Ideal Model: Secure Message Passing

(init) **to** TH **via** in_u?

- if $init_u = 1$ then abort
  (User u has already initiated key exchange.)
- $init_u := 1$
  (User u has initiated key exchange.)
- output (init, u) at $\mathsf{out_{sim}}!$ and schedule $\mathsf{out_{sim}}!$

(initok, u) **to** TH **via** $\mathsf{in_{sim}}$?

- if $init_1 = 0$ or $init_2 = 0$ or $key_u = 1$ then abort
  (One user has not initiated key exchange yet or user u has already received a key.)
- $key_u := 1$
  (The adversary has not disrupted key distribution to user u.)
- output $(\mathsf{initialized})$ at $\mathsf{out}_u!$ and schedule $\mathsf{out}_u!$

(send, m) **to** TH **via** in_1?

- if $key_1 \neq 1$ then abort
  (No key yet, hence no encryption possible.)
- if $sc \geq s$ or $\mathsf{len}(m) > L(k)$ then abort
  (Too many messages sent already or message too long.)
- $deliver[sc] := (sc, m)$
- $sc := sc + 1$
- output $(\mathsf{busy}, sc - 1, \mathsf{len}(m))$ at $\mathsf{out_{sim}}!$ and schedule $\mathsf{out_{sim}}!$

(select, i) **to** TH **via** $\mathsf{in_{sim}}$?

- if $key_2 \neq 1$ then abort
  (User 2 has not received a key yet, thus decryption not possible.)
- if $0 \leq i < sc (= \mathsf{size}(deliver))$ then output $(\mathsf{receive}, deliver[i])$ at $\mathsf{out}_2!$ and schedule $\mathsf{out}_2!$

3 The Hybrid Model

We now define a hybrid model that uses an actual encryption algorithm, but still relies on an ideal authentication subsystem to deliver messages. We will prove this hybrid model to be perfectly as secure as the ideal model in the black box simulatability sense [11]. As encryption primitive we will use the one time pad, but we will attempt to give a general formulation, so that the one time pad can easily be replaced with a different stream cipher. Of course, no more than computational security can be expected then.

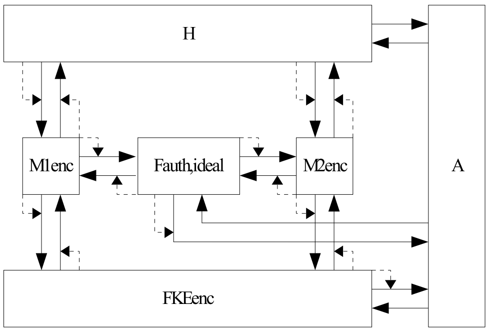

Figure 2: Hybrid Model: Secure Message Passing

The hybrid real model is sketched in Figure 2. The two machines $\mathsf{M}_{1,\mathsf{enc}}$, $\mathsf{M}_{2,\mathsf{enc}}$ handle encryption and decryption respectively. Between the two machines we still have an authenticated channel, implemented by the ideal authentication functionality $\mathsf{F}^{\mathsf{auth,ideal}}_{s,L_{\mathsf{auth}}}$. The maximal message length L_{auth} for the authentication subsystem is given as polynomial over $\mathbb{N}$ in k defined as $L_{\mathsf{auth}}(k) := L(k) + s \cdot k \geq L(k) + \lceil \log_2(s \cdot k) \rceil$. The authentication subsystem has to handle messages that are $\lceil \log_2(s \cdot k) \rceil$ longer than the messages handled by the encryption machines, because the current position within the one time pad key used for encryption has to be authenticated with the original message. With $\mathcal{H} = \{1, 2\}$ as above, the hybrid system is formally given as

$$Sys^{\mathsf{secmsg,hybrid}}_{s,L} = \{(\{\mathsf{M}_{u,\mathsf{enc}}, \mathsf{F}^{\mathsf{auth,ideal}}_{s,L_{\mathsf{auth}}}, \mathsf{FKE}_{\mathsf{enc}} | u \in \mathcal{H}\}, S)\} \tag{2}$$

where

$$\begin{aligned}
\mathsf{ports}(\mathsf{M}_{u,\mathsf{enc}}) = \{&\mathsf{in}_u?, \mathsf{out}_u!, \mathsf{out}^{\triangleleft}_u!, \mathsf{in}_{u,\mathsf{auth}}!, \mathsf{out}_{u,\mathsf{auth}}?, \mathsf{in}^{\triangleleft}_{u,\mathsf{auth}}!, \\
&\mathsf{in}_{u,\mathsf{FKE_{enc}}}!, \mathsf{in}^{\triangleleft}_{u,\mathsf{FKE_{enc}}}!, \mathsf{out}_{u,\mathsf{FKE_{enc}}}?\} \qquad \text{where } u \in \mathcal{H} && (3)\\
\mathsf{ports}(\mathsf{F}^{\mathsf{auth,ideal}}_{s,L_{\mathsf{auth}}}) = \{&\mathsf{in}_{u,\mathsf{auth}}?, \mathsf{out}_{u,\mathsf{auth}}!, \mathsf{out}^{\triangleleft}_{u,\mathsf{auth}}! | u \in \mathcal{H}\} && (4)\\
\cup \{&\mathsf{in}_{\mathsf{sim,auth}}?, \mathsf{out}_{\mathsf{sim,auth}}!, \mathsf{out}_{\mathsf{sim,auth}}{}^{\triangleleft}!\} \\
\mathsf{ports}(\mathsf{FKE}_{\mathsf{enc}}) = \{&\mathsf{in}_{u,\mathsf{FKE_{enc}}}?, \mathsf{out}_{u,\mathsf{FKE_{enc}}}!, \mathsf{out}^{\triangleleft}_{u,\mathsf{FKE_{enc}}}!, \\
&\mathsf{in}_{\mathsf{sim,FKE_{enc}}}?, \mathsf{out}_{\mathsf{sim,FKE_{enc}}}!, \mathsf{out}^{\triangleleft}_{\mathsf{sim,FKE_{enc}}}! | u \in \mathcal{H}\} && (5)
\end{aligned}$$

and the specified ports are given by $S^c := \{\mathsf{in}_u!, \mathsf{out}_u? | u \in \mathcal{H}\}$. The machines in $Sys^{\mathsf{secmsg,hybrid}}_{s,L}$ maintain the following data structures (where $u \in \mathcal{H}$):

$\mathsf{FKE}_{\mathsf{enc}}$**:** $init_u, distributed_u \in \{0, 1\}$, $key \in \Sigma^*$

$\mathsf{M}_{u,\mathsf{enc}}$**:** $init \in \{0, 1\}$, $enckey \in \Sigma^*$, $keypos \in \{0, \ldots, s \cdot L(k)\}$, $sc \in \{0, \ldots, s\}$

$\mathsf{F}^{\mathsf{auth,ideal}}_{s,L_{\mathsf{auth}}}$**:** $init_1, init_2, key_1, key_2 \in \{0, 1\}$, $sc \in \{0, \ldots, s\}$,
$deliver \in (\{0, \ldots, s\}, \Sigma^k)^*$

where all variables are initialized to 0, the empty list [] or the empty string ε as applicable. When operating on strings in Σ^* or lists, we let $x[a : b]$ denote the substring (in case x is a string) or sublist (in case x is a list) from (and including) position a up to (but not including) position b. The state-transition functions of the machines are given by the rules in the box below. Inputs not treated explicitly are ignored. If an input triggers a non-empty output, we say the machine accepts it. (Note that the rules are ordered by the machines they belong to and may be invoked by rules listed further down.)

(init) **to** $\mathsf{M}_{u,\mathsf{enc}}$ **via** $\mathsf{in}_u?$

- output (init) at $\mathsf{in}_{u,\mathsf{auth}}!$ and schedule $\mathsf{in}_{u,\mathsf{auth}}!$

(initialized) **to** $\mathsf{M}_{u,\mathsf{enc}}$ **via** $\mathsf{out}_{u,\mathsf{auth}}?$

- $i := s \cdot L(k)$
- output $(\mathsf{generate}, i)$ at $\mathsf{in}_{u,\mathsf{FKE_{enc}}}!$ and schedule $\mathsf{in}_{u,\mathsf{FKE_{enc}}}!$

$(\mathsf{cipher}, (cnt, (keypos, c)))$ **to** $\mathsf{M}_{2,\mathsf{enc}}$ **via** $\mathsf{out}_{2,\mathsf{auth}}?$

- if $initauth \neq 1$ or $initenc \neq 1$ then abort
  (No key yet, hence no decryption possible.)
- if $\mathsf{len}(c) > L(k)$ or $keypos + \mathsf{len}(c) > s \cdot L(k)$ then abort
  (Message too long or $keypos$ out of range.)
- $m := c \oplus enckey[keypos : keypos + \mathsf{len}(c)]$
- output $(\mathsf{receive}, cnt, m)$ at $\mathsf{out}_2!$ and schedule $\mathsf{out}_2!$

(init) **to** $\mathsf{F}^{\mathsf{auth,ideal}}_{s,L_{\mathsf{auth}}}$ **via** $\mathsf{in}_{u,\mathsf{auth}}?$

- if $init_u = 1$ then abort
  (User u has already initiated key exchange.)
- $init_u := 1$
  (User u has initiated key exchange.)
- output (init, u) at $\mathsf{out}_{\mathsf{sim,auth}}!$ and schedule $\mathsf{out}_{\mathsf{sim,auth}}!$

(initok, u) **to** $\mathsf{F}^{\mathsf{auth,ideal}}_{s,L_{\mathsf{auth}}}$ **via** $\mathsf{in}_{\mathsf{sim,auth}}?$

- if $init_1 = 0$ or $init_2 = 0$ or $key_u = 1$ then abort
  (One user has not initiated key exchange yet or user u has already received a key.)
- $key_u := 1$
  (The adversary has not disrupted the distribution of keys to user u.)
- output (initialized) at $\mathsf{out}_{u,\mathsf{auth}}!$ and schedule $\mathsf{out}_{u,\mathsf{auth}}!$

(key, key) **to** $\mathsf{M}_{u,\mathsf{enc}}$ **via** $\mathsf{out}_{u,\mathsf{FKE}_{\mathsf{enc}}}$?

- $enckey := key$
- $init := 1$
- output (initialized) at out_u! and schedule out_u!

(send, m) **to** $\mathsf{M}_{1,\mathsf{enc}}$ **via** in_1?

- if $init \neq 1$ then abort
  (No key yet, hence no encryption possible.)
- if $sc \geq s$ or $\mathsf{len}(m) > L(k)$ then abort
  (Too many messages sent already or message too long.)
- $sc := sc + 1$
- $c := (keypos, m \oplus enckey[keypos : keypos + \mathsf{len}(m)])$
  (For transmission we encode $keypos$ as $\lceil \log_2(s \cdot k) \rceil$ bit string.)
- $keypos := keypos + \mathsf{len}(m)$
- output (cipher, c) at $\mathsf{in}_{1,\mathsf{auth}}$! and schedule $\mathsf{in}_{1,\mathsf{auth}}$!

(cipher, c) **to** $\mathsf{F}^{\mathsf{auth,ideal}}_{s,L_{\mathsf{auth}}}$ **via** $\mathsf{in}_{1,\mathsf{auth}}$?

- if $key_1 \neq 1$ then abort
  (No key yet, hence no authentication possible.)
- if $sc \geq s$ or $\mathsf{len}(m) > L_{\mathsf{auth}}(k)$ then abort
  (Too many messages sent already or message too long.)
- $deliver[sc] := (sc, c)$
  (For transmission we encode sc as $\lceil \log_2 s \rceil$ bit string.)
- $sc := sc + 1$
- output (busy, $sc - 1, c$) at $\mathsf{out}_{\mathsf{sim,auth}}$! and schedule $\mathsf{out}_{\mathsf{sim,auth}}$!

(select, i) **to** $\mathsf{F}^{\mathsf{auth,ideal}}_{s,L}$ **via** $\mathsf{in}_{\mathsf{sim,auth}}$?

- if $key_2 \neq 1$ then abort
  (User 2 has not received a key yet, thus authentication not possible.)
- if $0 \leq i < sc$ ($= \mathsf{size}(deliver)$) then output (cipher, $deliver[i]$) at $\mathsf{out}_{2,\mathsf{auth}}$! and schedule $\mathsf{out}_{2,\mathsf{auth}}$!

(generate, i) **to** $\mathsf{FKE}_{\mathsf{enc}}$ **via** $\mathsf{in}_{u,\mathsf{FKE}_{\mathsf{enc}}}$?, $u \in \{1, 2\}$

- if $init_{3-u} = 1$ and $init_u = 0$ then
  - $init_u := 1$
  - $key \in \mathbb{F}_2^i$ drawn uniformly at random
  - output (generate, i) at $\mathsf{out}_{\mathsf{sim,FKE}_{\mathsf{enc}}}$! and schedule $\mathsf{out}_{\mathsf{sim,FKE}_{\mathsf{enc}}}$!

  else $init_u := 1$

(distribute, u) **to** $\mathsf{FKE}_{\mathsf{enc}}$ **via** $\mathsf{in}_{\mathsf{sim,FKE}_{\mathsf{enc}}}$?

- if $distributed_u = 1$ or $init_1 = 0$ or $init_2 = 0$ then abort
- $distributed_u := 1$
- output (key, key) at $\mathsf{out}_{u,\mathsf{FKE}_{\mathsf{enc}}}$! and schedule $\mathsf{out}_{u,\mathsf{FKE}_{\mathsf{enc}}}$!

Caveat 2 *The position in the key sequence must be included with the message unencrypted. It should be authenticated with the message, unless tampering with the key sequence position ensures (with overwhelming probability) that authentication will fail.*

Rational. If the key sequence position is not included, the adversary suppressing one single message results in loss of synchronization. Thus decryption will fail for all subsequent messages. If the key position is not authenticated, the adversary may modify it without being noticed. If the authentication is then not guaranteed to fail, the receiver will be under the impression he is receiving nonsensical messages from the sender. □

4 Black Box Perfect Indistinguishability of Hybrid & Ideal Model

We show now, that the ideal model depicted in Figure 1 and the hybrid real model as in Figure 2 are black box perfectly indistinguishable according to the definition set forth in [11]. That implies that the hybrid real model is perfectly at least as secure as the ideal model.

Theorem 1 *The hybrid real system* $Sys^{\mathsf{secmsg,hybrid}}_{s,L}$ *as in Figure 2 is black box perfectly at least as secure as* *the ideal system* $Sys^{\mathsf{secmsg,ideal}}_{s,L}$ *as in Figure 1:*

$$Sys^{\mathsf{secmsg,hybrid}}_{s,L} \geq^{f,\mathsf{perf}}_{\mathsf{sec}} Sys^{\mathsf{secmsg,ideal}}_{s,L} \tag{6}$$

where the valid mapping f between the systems is obvious from the machine names.

Proof. Given an adversary A in the hybrid real model we define the adversary A' for the ideal model using the simple black box construction shown in Figure 3. All machines are given exactly as in the hybrid-real model, with three exceptions. On the machines $\mathsf{M}_{u,\mathsf{enc}}$ the ports $\mathsf{in}_u?, \mathsf{out}_u!$ are renamed to $\mathsf{in}'_u?, \mathsf{out}'_u!$ and MUX is newly defined. MUX acts as multiplexer, that distributes the outputs of TH to the submachines of the simulator S. The ports of MUX are $\mathsf{ports(MUX)} = \{\mathsf{in_{sim}}!, \mathsf{out_{sim}}?, \mathsf{in}'_u!, \mathsf{out}'_u?|u \in \mathcal{H}\}$ and it operates as follows:

(init, u) **to** MUX **via** $\mathsf{out_{sim}}?$ • output (init) at $\mathsf{in}'_u!$ and schedule $\mathsf{in}'_u!$ $(\mathsf{initialized})$ **to** MUX **via** $\mathsf{out}'_u?$ • output (initok, u) at $\mathsf{in_{sim}}!$ and schedule $\mathsf{in_{sim}}?$	$(\mathsf{busy}, i, \mathsf{len}(m))$ **to** MUX **via** $\mathsf{out_{sim}}?$ • $m := 0^{\mathsf{len}(m)} \in \mathbb{F}_2^{\mathsf{len}(m)}$ • output (send, m) at $\mathsf{in}'_1!$ and schedule $\mathsf{in}'_1!$ $(\mathsf{receive}, cnt, m)$ **to** MUX **via** $\mathsf{out}'_2?$ • output (select, cnt) at $\mathsf{in_{sim}}!$ and schedule $\mathsf{in_{sim}}!$

To complete the proof that the hybrid real model is black box perfectly as secure as the ideal model we have to show, that under the given black box construction and for a fixed adversary A for both models, the views of the honest users H are perfectly indistinguishable.

But this is clear from the construction of the simulator and the fact, that the one time pad is perfectly concealing. The black box construction as given above clearly replicates the hybrid real protocol identically within the simulator S, but for the content of the messages. But the one time pad is perfectly concealing, and thus renders messages of different content indistinguishable. So there is no way for either adversary A or the users H to distinguish the ideal and the hybrid real structure. Thus the views for the honest users H are left perfectly indistinguishable and perfect black box simulatability is proven. □

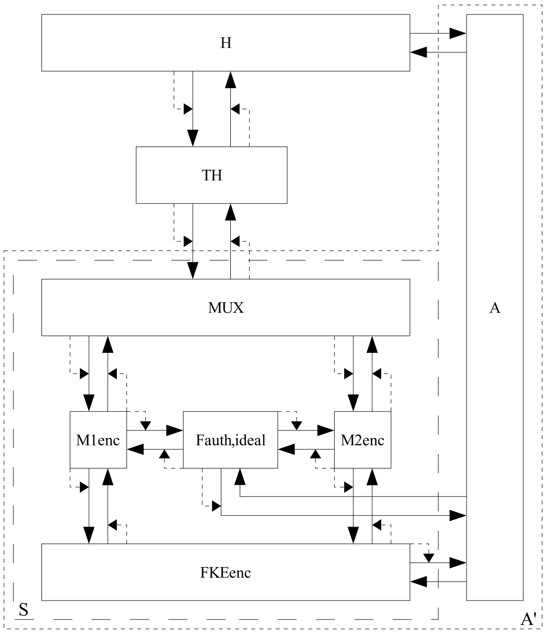

Figure 3: Black Box Construction

5 The Real Model

The real model defined in this section describes the actual usable protocol for statistically secure message passing. There are two machines M_1 and M_2 for sender and receiver respectively. Each machine M_u decomposes into two submachines $\mathsf{M}_{u,\mathsf{enc}}$ and $\mathsf{M}_{u,\mathsf{auth}}$ that handle encryption and authentication. The machines $\mathsf{M}_{u,\mathsf{auth}}$ and $\mathsf{FKE}_{\mathsf{auth}}$ constitute the real authentication subsystem $Sys_{s,L}^{\mathsf{auth,real}}$ that replaces the ideal authentication subsystem $Sys_{s,L}^{\mathsf{auth,ideal}}$. As such the real system is mostly identical to the hybrid-real model, only replacing the ideal authentication system with the real one, and thus allowing for a proof of security by composition.

The real authentication system is based on a statistically secure authentication procedure described in [14, Chapter 10.3]. It views the message m as number in $\mathbb{F}_p$ and uses an affine authentication function, computing the authentication tag as $t(m, k_1, k_2) := (m \cdot k_1 + k_2) \mod p$ where $k_1, k_2 \in \mathbb{F}_p$ is the current pair of keys. A more detailed discussion of this scheme and its security will be given in the next section.

With $\mathcal{H} = \{1, 2\}$ the real system is formally given as

$$Sys_{s,L}^{\mathsf{secmsg,real}} = \{(\{\mathsf{M}_{u,\mathsf{enc}}, \mathsf{M}_{u,\mathsf{auth}}, \mathsf{FKE}_{\mathsf{auth}}, \mathsf{FKE}_{\mathsf{enc}} | u \in \mathcal{H}\}, S)\} \tag{7}$$

where

$$\begin{aligned}
\mathsf{ports}(\mathsf{M}_{u,\mathsf{enc}}) &= \{\mathsf{in}_u?, \mathsf{out}_u!, \mathsf{out}_u^{\triangleleft}!, \mathsf{in}_{u,\mathsf{auth}}!, \mathsf{out}_{u,\mathsf{auth}}?, \mathsf{in}_{u,\mathsf{auth}}^{\triangleleft}!, \qquad (8)\\
&\quad \mathsf{in}_{u,\mathsf{FKE_{enc}}}!, \mathsf{in}_{u,\mathsf{FKE_{enc}}}^{\triangleleft}!, \mathsf{out}_{u,\mathsf{FKE_{enc}}}?\} \qquad \text{where } u \in \mathcal{H}\\
\mathsf{ports}(\mathsf{M}_{1,\mathsf{auth}}) &= \{\mathsf{in}_{1,\mathsf{auth}}?, \mathsf{out}_{1,\mathsf{auth}}!, \mathsf{out}_{1,\mathsf{auth}}^{\triangleleft}!, \mathsf{netout}!, \qquad (9)\\
&\quad \mathsf{in}_{1,\mathsf{FKE_{auth}}}!, \mathsf{in}_{1,\mathsf{FKE_{auth}}}^{\triangleleft}!, \mathsf{out}_{1,\mathsf{FKE_{auth}}}?\}\\
\mathsf{ports}(\mathsf{M}_{2,\mathsf{auth}}) &= \{\mathsf{in}_{2,\mathsf{auth}}?, \mathsf{out}_{2,\mathsf{auth}}!, \mathsf{out}_{2,\mathsf{auth}}^{\triangleleft}!, \mathsf{netin}?, \qquad (10)\\
&\quad \mathsf{in}_{2,\mathsf{FKE_{auth}}}!, \mathsf{in}_{2,\mathsf{FKE_{auth}}}^{\triangleleft}!, \mathsf{out}_{2,\mathsf{FKE_{auth}}}?\}\\
\mathsf{ports}(\mathsf{FKE}_{\mathsf{auth}}) &= \{\mathsf{in}_{u,\mathsf{FKE_{auth}}}?, \mathsf{out}_{u,\mathsf{FKE_{auth}}}!, \mathsf{out}_{u,\mathsf{FKE_{auth}}}^{\triangleleft}!, \qquad (11)\\
&\quad \mathsf{in}_{\mathsf{sim},\mathsf{FKE_{auth}}}?, \mathsf{out}_{\mathsf{sim},\mathsf{FKE_{auth}}}!, \mathsf{out}_{\mathsf{sim},\mathsf{FKE_{auth}}}^{\triangleleft}! | u \in \mathcal{H}\}\\
\mathsf{ports}(\mathsf{FKE}_{\mathsf{enc}}) &= \{\mathsf{in}_{u,\mathsf{FKE_{enc}}}?, \mathsf{out}_{u,\mathsf{FKE_{enc}}}!, \mathsf{out}_{u,\mathsf{FKE_{enc}}}^{\triangleleft}!, \qquad (12)\\
&\quad \mathsf{in}_{\mathsf{sim},\mathsf{FKE_{enc}}}?, \mathsf{out}_{\mathsf{sim},\mathsf{FKE_{enc}}}!, \mathsf{out}_{\mathsf{sim},\mathsf{FKE_{enc}}}^{\triangleleft}! | u \in \mathcal{H}\}
\end{aligned}$$

and the specified ports are given by $S^c := \{\mathsf{in}_u!, \mathsf{out}_u? | u \in \mathcal{H}\}$.

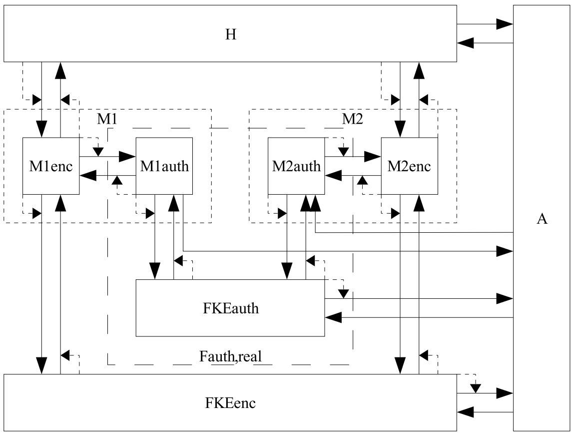

Figure 4: Real Model: Application of One Time Pad

The machines in $Sys_{s,L}^{\mathsf{secmsg,real}}$ maintain the following data structures (where $u \in \mathcal{H}$ and $v \in \{enc, auth\}$):

$\mathsf{FKE}_{\mathsf{enc}}$: $init_u, distributed_u \in \{0, 1\}$, $key \in \Sigma^*$

$\mathsf{FKE}_{\mathsf{auth}}$: $init_u, distributed_u \in \{0, 1\}$, $key \in \Sigma^*$, $p \in \{0, \dots, 2^{k+1} - 1\}$

$\mathsf{M}_{u,\mathsf{enc}}$: $init \in \{0, 1\}$, $enckey \in \Sigma^*$, $keypos \in \{0, \dots, s \cdot L(k)\}$, $sc \in \{0, \dots, s\}$

$\mathsf{M}_{u,\mathsf{auth}}$: $initauth \in \{0, 1\}$, $authkey \in \Sigma^*$, $sc \in \{0, \dots, s\}$, $p \in \{0, \dots, 2^{k+1} - 1\}$

where all variables are initialized to $0, [], \varepsilon$ as applicable. Again, all differences to the hybrid real model are confined to the authentication subsystem.

The state-transition functions of all machines are given as in the hybrid real model, except for the machines that belong to the real authentication subsystem. For those the state transition functions are described below. Again, inputs not treated explicitly are ignored.

(init) **to** $\mathsf{M}_{u,\mathsf{auth}}$ **via** $\mathsf{in}_{u,\mathsf{auth}}?$

- $j := \max\{k+1, L_{\mathsf{auth}}(k)+2\}$
- output $(\mathsf{generate}, s, j)$ at $\mathsf{in}_{u,\mathsf{FKE}_{\mathsf{auth}}}!$ and schedule $\mathsf{in}_{u,\mathsf{FKE}_{\mathsf{auth}}}!$

(cipher, c) **to** $\mathsf{M}_{1,\mathsf{auth}}$ **via** $\mathsf{in}_{1,\mathsf{auth}}?$

- if $initauth \neq 1$ then abort
  (No key yet, hence no authentication possible.)
- if $sc \geq s$ or $\mathsf{len}(c) > L_{\mathsf{auth}}(k)$ then abort
  (Too many messages sent already or message too long.)
- $m := 1||c$
  (Prefix the bitstring c with a leading one.)
- $tag := (m \cdot key[2 \cdot sc] + key[2 \cdot sc+1]) \mod p$
  (Calculating the authentication tag we view m as number in $\mathbb{Z}$.)
- $pkt := (sc, m, tag)$
- $sc := sc + 1$
- output (packet, pkt) at $\mathsf{netout}!$

$(\mathsf{key}, key, modulus)$ **to** $\mathsf{M}_{u,\mathsf{auth}}$ **via** $\mathsf{out}_{u,\mathsf{FKE}_{\mathsf{auth}}}?$

- $p := modulus$
- $authkey := key$
- $initauth := 1$
- output $(\mathsf{initialized})$ at $\mathsf{out}_{u,\mathsf{auth}}!$ and schedule $\mathsf{out}_{u,\mathsf{auth}}!$

(packet, pkt) **to** $\mathsf{M}_{2,\mathsf{auth}}$ **via** $\mathsf{netin}?$

- if $initauth \neq 1$ then abort
  (No key yet, hence no authentication possible.)
- $(pos, m, tag) := pkt$
  (As pos, tag are bitstrings of fixed length, segmentation is easy)
- if $m[0] \neq 1$ or $pos \geq s$ or $\mathsf{len}(m) - 1 > L_{\mathsf{auth}}(k)$ then abort
  (Leading bit wrong, too many messages sent or message too long.)
- if $tag \neq (m \cdot key[2 \cdot pos] + key[2 \cdot pos + 1]) \mod p$ then abort
  (Authentication failed.)
- output $(\mathsf{cipher}, pos, m[1 : \mathsf{len}(m)])$ at $\mathsf{out}_{2,\mathsf{auth}}!$ and schedule $\mathsf{out}_{2,\mathsf{auth}}!$

$(\mathsf{generate}, i, j)$ **to** $\mathsf{FKE}_{\mathsf{auth}}$ **via** $\mathsf{in}_{u,\mathsf{FKE}_{\mathsf{auth}}}?$, $u \in \{1,2\}$

- if $init_{3-u} = 1$ and $init_u = 0$ then
  - choose $p \in \{2^{j-1} \leq q < 2^j : q \text{ prime}\}$ arbitrary
    (Choose a j bit prime.)
  - $init_u := 1$
  - $key := []$
  - while size$(key) < 2i$
    - $a \in \mathbb{F}_2^j$ drawn uniformly at random
    - if $a < p$ then $key := key||a$
      (append a to list key)
  - output $(\mathsf{generate}, i, p)$ at $\mathsf{out}_{\mathsf{sim},\mathsf{FKE}_{\mathsf{auth}}}!$ and schedule $\mathsf{out}_{\mathsf{sim},\mathsf{FKE}_{\mathsf{auth}}}!$

  else $init_u := 1$

$(\mathsf{distribute}, u)$ **to** $\mathsf{FKE}_{\mathsf{auth}}$ **via** $\mathsf{in}_{\mathsf{sim},\mathsf{FKE}_{\mathsf{auth}}}?$

- if $distributed_u = 1$ or $init_1 = 0$ or $init_2 = 0$ then abort
- $distributed_u := 1$
- output (key, key, p) at $\mathsf{out}_{u,\mathsf{FKE}_{\mathsf{auth}}}!$ and schedule $\mathsf{out}_{u,\mathsf{FKE}_{\mathsf{auth}}}!$

Caveat 3 *Our authentication scheme does not protect leading zeros, since m and $0||m$ correspond to the same number in $\mathbb{F}_p$. Therefore, we make sure, that every message starts with a one.*

Caveat 4 *We need to include the message sequence number with the authenticated message, since loss of synchronization would otherwise prevent us from authenticating messages after the adversary has suppressed one. The message sequence number need not be authenticated, since modification of the sequence number will just lead to failing authentication.*

It remains to show that $Sys^{\mathsf{secmsg,real}}_{s,L}$ is black box statistically as secure as $Sys^{\mathsf{secmsg,hybrid}}_{s,L}$. This is done by composition. We will prove that the real authentication subsystem $Sys^{\mathsf{auth,real}}_{s,L}$ is black box statistically as secure as the ideal authentication subsystem $Sys^{\mathsf{auth,ideal}}_{s,L}$ utilized in $Sys^{\mathsf{secmsg,real}}_{s,L}$. The statistical black box security of $Sys^{\mathsf{secmsg,ideal}}_{s,L}$ then follows from the perfect black box security of $Sys^{\mathsf{secmsg,hybrid}}_{s,L}$ using the composition theorem from [11].

6 Security of the Authentication Subsystem

The ideal authentication subsystem

$$Sys^{\mathsf{auth,ideal}}_{s,L_{\mathsf{auth}}} = \{(\{\mathsf{F}^{\mathsf{auth,ideal}}_{s,L_{\mathsf{auth}}}\}, S)\}, \tag{13}$$

utilizing the ideal authentication functionality (structure)

$$F^{\mathsf{auth,ideal}}_{s,L_{\mathsf{auth}}} = (\{\mathsf{F}^{\mathsf{auth,ideal}}_{s,L_{\mathsf{auth}}}\}, S) \tag{14}$$

is depicted in Figure 5. The real authentication subsystem

$$Sys^{\mathsf{auth,real}}_{s,L_{\mathsf{auth}}} = \{(\{\mathsf{M}_{u,\mathsf{auth}}, \mathsf{FKE}_{\mathsf{auth}} | u \in \mathcal{H}\}, S)\} \tag{15}$$

utilizing the real authentication functionality

$$F^{\mathsf{auth,real}}_{s,L_{\mathsf{auth}}} = (\{\mathsf{M}_{u,\mathsf{auth}}, \mathsf{FKE}_{\mathsf{auth}} | u \in \mathcal{H}\}, S) \tag{16}$$

is shown in Figure 6. All machine definitions and the trust model are as given above and the specified ports are given by $S^c := \{\mathsf{in}_{u,\mathsf{auth}}!, \mathsf{out}_{u,\mathsf{auth}}? | u \in \mathcal{H}\}$ where of course $\mathcal{H} = \{1, 2\}$.

Theorem 2 *The real authentication subsystem $Sys^{\mathsf{auth,real}}_{s,L_{\mathsf{auth}}}$ as in Figure 6 is black box statistically at least as secure as the ideal authentication subsystem $Sys^{\mathsf{auth,ideal}}_{s,L_{\mathsf{auth}}}$ as in Figure 5: $Sys^{\mathsf{auth,real}}_{s,L_{\mathsf{auth}}} \geq^{f,\mathrm{ExpSmall}}_{\mathsf{sec}} Sys^{\mathsf{auth,ideal}}_{s,L_{\mathsf{auth}}}$ where the valid mapping f between the systems is obvious from the machine names.*

Proof. It is clear from its definition that the ideal authentication functionality $F^{\mathsf{auth,ideal}}_{s,L_{\mathsf{auth}}}$ does guarantee authenticated transmission. We will now show that the real authentication functionality $F^{\mathsf{auth,real}}_{s,L_{\mathsf{auth}}}$, that transmits all data via the adversary A, is black box statistically as secure as the ideal functionality.

To this end, given an adversary A in the real model we define the adversary A' for the ideal model using the black box simulator Sim as shown in Figure 7. All machines are given exactly as in the real model, with three exceptions. On the machines $\mathsf{M}_{u,\mathsf{auth}}$ the ports $\mathsf{in}_{u,\mathsf{auth}}?, \mathsf{out}_{u,\mathsf{auth}}!$ are renamed to $\mathsf{in}'_{u,\mathsf{auth}}?, \mathsf{out}'_{u,\mathsf{auth}}!$ and MUX' is newly defined. MUX' acts as multiplexer, that distributes the outputs of $\mathsf{F}^{\mathsf{auth,ideal}}_{s,L_{\mathsf{auth}}}$ to the submachines of the simulator Sim. The ports of MUX' are $\mathsf{ports}(\mathsf{MUX}') = \{\mathsf{in}_{\mathsf{sim,auth}}!, \mathsf{out}_{\mathsf{sim,auth}}?, \mathsf{in}'_{u,\mathsf{auth}}!, \mathsf{out}'_{u,\mathsf{auth}}? | u \in \mathcal{H}\}$ and it operates as follows:

(init, u) **to** MUX' **via** $\mathsf{out}_{\mathsf{sim,auth}}?$ • output (init) at $\mathsf{in}'_{u,\mathsf{auth}}!$ and schedule $\mathsf{in}'_{u,\mathsf{auth}}!$ $(\mathsf{initialized})$ **to** MUX' **via** $\mathsf{out}'_{u,\mathsf{auth}}?$ • output (initok, u) at $\mathsf{in}_{\mathsf{sim,auth}}!$ and schedule $\mathsf{in}_{\mathsf{sim,auth}}!$	(busy, i, c) **to** MUX' **via** $\mathsf{out}_{\mathsf{sim,auth}}?$ • output (cipher, c) at $\mathsf{in}'_{1,\mathsf{auth}}!$ and schedule $\mathsf{in}'_{1,\mathsf{auth}}!$ $(\mathsf{cipher}, (cnt, c))$ **to** MUX' **via** $\mathsf{out}'_2?$ • output (select, cnt) at $\mathsf{in}_{\mathsf{sim,auth}}!$ and schedule $\mathsf{in}_{\mathsf{sim,auth}}!$

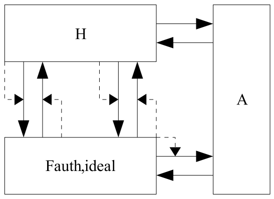

Figure 5: Ideal Model: Authentication

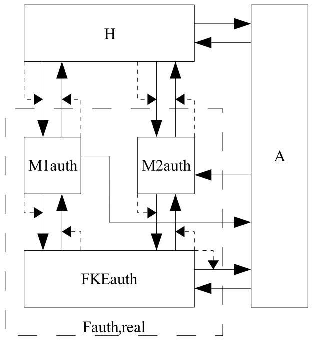

Figure 6: Real Model: Authentication

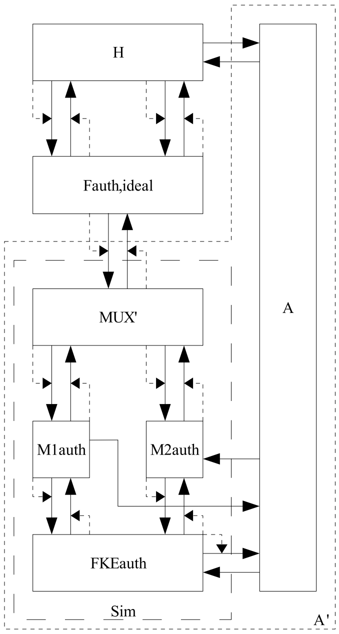

Figure 7: Black Box Simulator: Authentication

To complete the proof that the real authentication system is black box statistically as secure as the ideal system we have to show, that, under the given black box construction and for a fixed adversary A for both models, the views of the honest users H are statistically indistinguishable.

The black box construction as given above clearly replicates the real authentication protocol identically within the simulator Sim. So there is no way for the adversary A to distinguish the ideal and the real structure. Thus the views for the honest users H are left identical unless A manages to introduce forged messages.

Hence, we take a look at the probability that A successfully introduces a forged message m'. First we note that the probability $P(t(m,k_1,k_2)|m)$ of a certain tag $t(m,k_1,k_2) := (m \cdot k_1 + k_2) \mod p$ given a message m assuming a uniform distribution over the keys $k_1, k_2 \in \mathbb{F}_p$ is $P(t|m) = \frac{1}{p}$.

Therefore, the adversary A has a chance of exactly $\frac{1}{p} \leq \frac{1}{2^k}$, i.e. exponentially small in the security parameter k, to successfully introduce a forged message and tag (m', t'). That holds, even if the adversary A has intercepted the current message m, since for $m \neq m'$ and an arbitrary choice of the forged tag t' we have $P[t' = t(m', k_1, k_2)|(m, t(m, k_1, k_2))] = \frac{1}{p}$ (see [14, Chapter 10.3]).

If we consider a polynomial $l(X)$ and perform $l(k)$ many authentications for security parameter k, then we find that the probability for a successful deception is bounded by $l(k)\frac{1}{2^k} \in \text{ExpSmall}(k)$. That means, if we consider a configuration of the real authentication system $conf_{\mathsf{real}} = (F^{\mathsf{auth,real}}_{s,L_{\mathsf{auth}}}, S_{\mathsf{auth}}, \mathsf{H}, \mathsf{A}) \in \mathsf{Conf}(Sys^{\mathsf{auth,real}}_{s,L_{\mathsf{auth}}})$ and the corresponding configuration $conf_{\mathsf{ideal}} = (F^{\mathsf{auth,ideal}}_{s,L_{\mathsf{auth}}}, S_{\mathsf{auth}}, \mathsf{H}, \mathsf{A}') \in \mathsf{Conf}(Sys^{\mathsf{auth,ideal}}_{s,L_{\mathsf{auth}}})$ of the ideal system, we obtain views for the honest users H, such that $view_{conf_{\mathsf{ideal}},l}(\mathsf{H}) \approx_{\text{ExpSmall}} view_{conf_{\mathsf{real}},l}(\mathsf{H})$ where $view_{conf_{\mathsf{ideal}},l}(\mathsf{H})$, $view_{conf_{\mathsf{real}},l}(\mathsf{H})$ are the families of l-step prefixes of the views for arbitrary polynomials l and the valid mapping $f : Sys^{\mathsf{auth,real}}_{s,L_{\mathsf{auth}}} \longrightarrow Sys^{\mathsf{auth,ideal}}_{s,L_{\mathsf{auth}}}$ is clear. We may therefore conclude that by the definitions in [11] the real authentication system is statistically at least as secure as the ideal system: $Sys^{\mathsf{auth,real}}_{s,L_{\mathsf{auth}}} \geq^{f,\text{ExpSmall}}_{\mathsf{sec}} Sys^{\mathsf{auth,ideal}}_{s,L_{\mathsf{auth}}}$ □

Caveat 5 *The statistical security of the authentication scheme given here is only guaranteed, as long as the message m interpreted as a natural number is bounded by the modulus p.*

Rational. If we allowed messages $m \geq p$ the adversary could easily introduce a forged message $(m+p) \mod p$. This would go unnoticed, as $t(m+p, k_1, k_2) = ((m+p)k_1 + k_2) \mod p = (mk_1 + k_2) \mod p = t(m, k_1, k_2)$ □

Our system takes this into account by limiting the message length to at most $L_{\mathsf{auth}}(k) + 1$ bits (including the leading one) and choosing p as $L_{\mathsf{auth}}(k) + 2$ bit prime (or larger).

The composition theorem of [11, Theorem 4.1] is applicable to the systems $Sys^{\mathsf{secmsg,hybrid}}_{s,L}$ and $Sys^{\mathsf{secmsg,real}}_{s,L}$ with the respective subsystems $Sys^{\mathsf{auth,ideal}}_{s,L_{\mathsf{auth}}}$ and $Sys^{\mathsf{auth,real}}_{s,L_{\mathsf{auth}}}$, since each system is composed of only one single structure and because the consistency condition on the ports is clearly fulfilled. Thus we have

Theorem 3 *The real system for secure message passing as given above is black box statistically as secure as the ideal system:* $Sys_{s,L}^{\mathsf{secmsg,real}} \geq_{\mathsf{sec}}^{f,\mathrm{ExpSmall}} Sys_{s,L}^{\mathsf{secmsg,ideal}}$ *where the valid mapping f between the systems is obvious.*

Note that the theorem above still holds if we replace the authentication system with any other statistically secure and composable authentication system.

7 Conclusion

We have seen, that it is feasible, but not trivial, to use the one time pad to construct a statistically secure message passing system. In particular we note, that it is impossible to obtain a perfectly secure system (i.e. perfectly indistinguishable from the ideal system), because authentication can at best be statistically secure.

The proof we presented is modular in the sense that it admits any choice of statistically secure authentication system. As indicated in Appendix A it is also easily extensible to computationally secure ciphers and authentication systems, thus providing a framework for statements about stream ciphers in the model of [11, 2, 3].

Acknowledgements. This research was partially funded by the project ProSecCo of the IST-FET programme of the EC and performed in cooperation with SECOQC. We thank Dennis Hofheinz and Dominique Unruh for helpful comments and discussions.

References

[1] M. Agrawal, N. Kayal, and N. Saxena. PRIMES is in P, 2002. `http://www.cse.iitk.ac.in/news/primality.html`.

[2] M. Backes. *Cryptographically Sound Analysis of Security Protocols*. PhD thesis, Universität des Saarlandes, Saarbrücken, Germany, 2002.

[3] M. Backes, B. Pfitzmann, and M. Waidner. Secure Asynchronous Reactive Systems. Cryptology ePrint Archive, Report 2004/082, 2004. `http://eprint.iacr.org/2004/082/`.

[4] M. Ben-Or, M. Horodecki, D. Mayers, D. Leung, and J. Oppenheim. Composability of quantum key distribution. Presented at a number of conferences including QIP 2004, January 2004.

[5] C. H. Bennett and G. Brassard. Quantum cryptography: Public-key distribution and coin tossing. In *Proceedings of IEEE International Conference on Computers, Systems and Signal Processing, Bangalore, India*, pages 175–179, Los Alamitos, California, USA, 1984. IEEE Press.

[6] Colin Boyd and Anish Mathuria. *Protocols for Authentication and Key Establishment*. Information Security and Cryptography. Springer, 2003.

[7] Ran Canetti and Hugo Krawczyk. Universally Composable Notions of Key Exchange and Secure Channels. Cryptology ePrint Archive, Report 2002/059, 2002. `http://eprint.iacr.org/2002/059/`. Extended version of [8].

[8] Ran Canetti and Hugo Krawczyk. Universally Composable Notions of Key Exchange and Secure Channels. In Lars Knudsen, editor, *Advances in Cryptology – EUROCRYPY 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 337–351. Springer, 2002.

[9] H. Krawczyk. The Order of Encryption and Authentication for Protecting Communications (or: How Secure is SSL?). In J. Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *LNCS*, pages 310–331. Springer, 2001.

[10] A. Menezes, P. C. van Oorschot, and S. Vanstone. *Handbook of Applied Cryptography*. CRC Press, Boca Raton, FL, USA, 1996.

[11] B. Pfitzmann and M. Waidner. A Model for Asynchronous Reactive Systems and its Application to Secure Message Transmission. Cryptology ePrint Archive, Report 2000/066, 2000. `http://eprint.iacr.org/2000/066/`.

[12] A. Poppe, A. Fedrizzi, T. Loruenser, O. Maurhardt, R. Ursin, H. R. Boehm, M. Peev, M. Suda, C. Kurtsiefer, H. Weinfurter, T. Jennewein, and A. Zeilinger. Practical Quantum Key Distribution with Polarization-Entangled Photons. lanl.arXiv.org e-Print archive, quant-ph/0404115, April 2004.

[13] R. Renner and R. Koenig. Universally composable privacy amplification against quantum adversaries. lanl.arXiv.org e-Print archive, quant-ph/0403133, March 2004.

[14] D. R. Stinson. *Cryptography – Theory and Practice.* CRC Press, Boca Raton, FL, USA, 1995.

A Using Computationally Secure Components

The statistically secure protocol for message passing given above was designed to be as simple, modular, and extensible as possible. The authentication subsystem may clearly be replaced with any other composable and statistically secure authentication system and as implementation of the key exchange functionality FKE we may use any type of statistically secure key exchange, for instance passing the key by hand or using a quantum key exchange according to [5] (for the composability see [4, 13]).

For some applications it is desirable, that messages arriving "late", i.e. after a message sent later than the one in question, be discarded, to ensure messages always arrive in order. Given the system above, guaranteeing the order of messages is simple. The secure message passing system $Sys^{\mathsf{secmsg,ideal}}_{s,L}$ as described above already passes message sequence numbers to the recipient. Ensuring that messages arrive in order reduces to the simple matter of adding an additional machine, that ensures the strict monotonicity of these sequence numbers.

Using the framework above with only computationally secure components (a case discussed, e. g., by Canetti and Krawczyk [7]), requires a little additional work:

Theorem 4 *All machines in the (real/hybrid-real/ideal) model as given above can be instantiated (as turing machines) in such a fashion, that every invocation of a machine will terminate after time polynomial in the security parameter k.*

Proof. Clearly, the machines $\mathsf{TH}, \mathsf{M}_{u,\mathsf{enc}}, \mathsf{M}_{u,\mathsf{auth}}, \mathsf{FKE}_{\mathsf{enc}}, \mathsf{F}^{\mathsf{auth,ideal}}_{s,L}$ will run for time at most polynomial in k on every invocation in the model, provided we implement the actual turing machines sensibly. In particular, no machine needs to read more than $(2s+1)\cdot(L(k)+s\cdot k+2)+\mathit{const}$ bits of input per invocation, if the length checks are performed properly. Here *const* is a constant overhead induced by the message labels (i.e init or packet).

For the machine $\mathsf{FKE}_{\mathsf{auth}}$ we need that it is feasible to calculate an arbitrary $j := L_{\mathsf{auth}}(k)+2$ bit prime $p \in \{2^{j-1} \leq q < 2^j : q \text{ prime}\}$ with negligible failure probability in time polynomial in k. Since testing a j bit number for primality is polynomial in j [1], it follows from the prime number theorem, as discussed in [10], that guessing and testing a j bit prime is possible in time polynomial in j with negligible failure probability and j is a polynomial in k. $\mathsf{FKE}_{\mathsf{auth}}$ then runs in time polynomial in k with negligible failure probability. □

Since every invocation of a machine in our models runs in time polynomial in k it is not hard to turn all machines into polynomial machines as specified by [11]. We simply have to enforce polynomial bounds on the total runtime of every machine. To this end every machine rejects input after a polynomial number of invocations by setting its length function [2] to zero.

Therefore the model given above can easily be modified to prove results on stream ciphers and authentication schemes, that offer only polynomial security.