UC framework for anonymous communication

István Vajda
Technical University of Budapest
vajda@hit.bme.hu
December 16, 2011

Abstract
In this research report we present an UC framework for the general task of anonymous communication. Definition of the ideal and the real models are carried out in the BPW (Backes-Pfitzmann-Waidner) formalism. It is shown how this approach relates to and extends earlier proposals [10],[15]. We consider also the adaptive adversary. An example is given for a wireless application.

1. Introduction

Anonymity is a property of network security. Various notions of anonymous communication exist and various attacker models have been considered. Anonymity service is required in several different tasks, such as anonymous message sending (anonymous communication)**,** anonymous authentication, anonymous key exchange and location privacy [1],[3-5],[7-8],[11-12],[14].

A. Pfitzmann [12] proposed a set of working definitions for anonymity, unlinkability, unobservability and pseudonimity. Basic notions, which we use also in this report, are defined in [12] as follows:

Sender anonymity is the property that a particular message is not linkable to any sender and that to a particular sender, no message is linkable.
Unlinkability of two or more items means that within this system, these items are no more and no less related than they are related concerning the a priori knowledge.
Receiver anonymity is the property that a particular message cannot be linked to any recipient and that to a particular recipient, no message is linkable.
Relationship anonymity means that it may be traceable who sends which messages and it may also be possible to trace who receives which messages, but it is untraceable, who communicates to whom. In other words, sender and recipient (or recipients in case of multicast) are unlinkable.
Relationship anonymity is a weaker property than each of sender anonymity and recipient anonymity: relationship anonymity is implied by each of the latter properties. Though unlinkability is a wider notion, below we will use it only in the narrower sense of relationship anonymity.

Though the above definitions grasp the essence of a notion, these definitions are informal. We have to specify that for/against which participant an anonymity property is guaranteed (honest sender/receiver, adversary). Furthermore, we want to state anonymity property about a system and not about "a particular message", i.e. we have to define the view, the set of potentially anonymity-related information seen/collected during a run, according to which the participant in regard experiences anonymity. We also have to define a priori knowledge, where a priori means the

knowledge of the adversary before running the system. A closely related issue is the definition of the set of tolerable impairments, i.e. the set of anonymity-related information which is leaked to the adversary from feasibility reasons. Indeed, the anonymity problem is very special with respect to tolerable impairments because here tolerable impairments are essentially related to the characteristics of the transmitted message flows and this fact has direct impact on anonymity definition also in the simulatability approach.

For instance, if actually there is only a single active sender and its transmission cannot be hidden (tolerable impairment) then sender-anonymity, obviously, cannot be guaranteed for the message on the corresponding communication link and without dummy packets it cannot be guaranteed at any other point of the system observable by the adversary. Indeed, the anonymity requirement must be coherent with the set of tolerable imperfections.

When we consider the realization of an anonymity service in a specific communication environment, first of all, we have to know what kind of anonymity-related information we have to protect. Such information may be dispersed in several layers of the communication stack, e.g. MAC address in data link layer, IP address in network layer or IDs carried by the payload in application layer. Honest protocols, usually running in the application layer have no access to contents belonging to lower layers. For manipulation of lower layer information with the aim of anonymization, additional special purpose HW/SW components are needed, e.g. a mixing network. An adversary scanning the communication links has access to information in all layers by having access to the complete packet. A wired link discovers the identity of the machines at its ending points. In contrary, a capability of tapping a wireless communication channel does not necessarily implies the capability of identification of the communication parties.

There are primary sources of anonymity-related information such as identifiers, keys, sender/link specific payload, and there are secondary sources, typically, traffic characteristics (e.g. packet length, amount of traffic).

In traditional approaches, protocols were defined for isolated execution. However, recently most of cryptographic systems consist of many protocols and thus these protocols must run in a complex environment where the protocols may run simultaneously or concurrently. Analysis of protocols by the traditional approach has become virtually impossible. Even if every component protocol is secure by itself, the security of the compound protocol must be analyzed anew. In contrary, in the UC framework, we analyze each protocol in isolation, and the protocol maintains its security when it is running within an environment wherein another protocol may be running concurrently. The framework guaratees that the compound protocol consisting of secure protocols becomes also secure. [15-19]

Canetti [6] and independently Pfitzmann and Waidner [13] proposed UC security frameworks (there is a third researcher, Backes, joined to the latter pair in several publications; therefore we will call this latter framework as Backes-Pfitzmann-

Waidner (BPW) framework). We use the latter framework. BPW provides a cryptographically sound, reactive Dolev-Yao type abstraction.

Since Canetti [6] proposed to work out UC framework among other cryptographic tasks and concerns also for anonymity, just a few publications appeared in this field [15],[11],[10],[8]. Wikström [15] published the first provably secure mix-net. Hevia [10] proposed a definition of secure anonymous communication based on the notion of indistinguishability formalism, similar to the approach used for semantically secure encryption schemes. Neither of these publications gave a comprehensive formulation for anonymous communication within UC framework. We tried to fill this gap in this research report by looking at the general task of anonymous communication, giving the definition of the corresponding ideal functionality and placing the latter in a system which serves also as a proof framework for anonymity protocols.

2. Related works

Wikström [15] published the first UC result for mix-nets. Wikström's ideal functionality F_{MN} is reproduced in the Appendix. Ideal functionality F_{MN} runs with sending machines $P_1,...,P_N$, mix-servers $M_1,...,M_k$ and ideal adversary A. The adversary is static, i.e. the adversary is allowed to corrupt servers before running the system. The adversary knows the

- identifier of sending participants
- identifier of running mix-servers
- list of output messages

Ideal functionality F_{MN} stops, when the number of mix-servers, which have been running since the start of the run becomes at least $k/2$.
For an input (in time order): $(P_{i_1},m_{i_1}),...,(P_{i_N},m_{i_N})$ the ideal functionality F_{MN} sends output $L'=(m_{j_1},...,m_{j_N})$ via all mix-servers, where list L' is obtained from $L=(m_{i_1},...,m_{i_N})$ by lexicographical ordering. The adversary knows a pair of lists $L''=(P_{i_1},...,P_{i_N})$, $L'=(m_{j_1},...,m_{j_N})$, without knowing the correspondence of list elements.

Wikström showed a protocol that UC-realizes these functionalities in a hybrid model with respect to static adversary that corrupts less than $k/2$ mix-servers under the DDH-assumption. The ideal subprotocols in the hybrid were the following: bulletin board (anyone can read messages written on the board, but cannot rewrite or delete them); distributed ElGamal key generation (generation of keys that are needed to shuffle and recover messages); two zero knowledge proof of relations (a relation between plaintexts and ciphertexts and a relation between ciphertexts before shuffling and after shuffling).

Definition of zero knowledge property is also based on the simulation paradigm. In fact, the simulation paradigm was first proposed in the context of zero knowledge. There exist general composability theorems, which support the proof technology (proof for hybrid models) exploited also by Wikström. This approach is essentially of

flavor of reduction techniques enhanced with composability. Paper [15] assumes static environment, it is essentially restricted to the use of known building blocks in construction of hybrids, because zero knowledge proofs for general problems, are typically very hard to give.

The approach we "promote" is more flexible. We define a generic ideal system. In particular, ideal functionality F_{MN} is a specific version of generic ideal system F_{acom} as a version. Our approach provides a proof system. The real protocol is transformed into a symbolic protocol. In this step of abstraction the real cryptographic primitives (and randomness) are eliminated by using composable cryptolibrary [2]. The fulfillment of the security requirement is proved for the symbolic protocol ideally by automatic proof system or by hand for smaller protocols.

Hevia [10] proposed a computational indistinguishability approach, similar to the definition of semantic encryption schemes, in order to give a strong definition for anonymity under computational constraint:
"…the adversary produces two message matrices (which encode message senders and receivers in a standard way), and it is allowed to passively observe the execution of a communication protocol under a random one of these two matrices and then is required to have non-negligible advantage in determining under which of the two matrices the protocol was executed."
We prove that our ideal system F_{acom} provides an equivalently strong definition for anonymity with the significant advantage that our ideal system is also part of a proof system for assessment of anonymity provided by different realizations. We extend the results to adaptive adversary.

An anonymity definition should express that no anonymity-related information can be obtained from viewing the run of an anonymizer in excess to the information known a priori. Indeed, we meet this requirement under computational constraint. Note, when we introduce an ideal model for anonymous communication and show for a protocol that it UC-realizes the ideal model we accept this wished kind of anonymity definition. Indeed, in an ideal system an adversary gains no information in excess to a piori, because we require that the distribution of the view for real world adversary cannot be distinguished from the one of the ideal adversary.

Le [11] proposes anonymity protocols within an UC approach for a wireless application (client-server RFID). They also propose realization which uses a PRF shared by all clients and the server. Furthermore, the proof (simulation) assumes a truly random function. In contrary, in our proof system we do not assume such trusted setup and only the standard model of cryptography is assumed. We show an example from wireless applications and we briefly analyze a Group-based Private Authentication protocol [1].

Though less related to our approach, however, because it is in the stream of formal analysis approaches for anonymity protocols, we mention also result [8], which presents a framework for verifying a variety of information hiding properties, using modal logic. Their adversarial model is very restricted compared to our need. The authors also admit that automatic tools are missing which are needed to increase the usability of their approach.

In a recent comprehensive review on research in the field of anonymous communication ([8]) the authors summarize the following way:
"Despite all security reports, an upper limit on the anonymity that a system can provide is given by *black box attacks*… anonymous communication can be secured only tactically (for short periods) and not strategically or in the long term."
In fact, current solutions are vulnerable against attackers who have access to both ends of the communication. A known example is the HTTP traffic where it is hard to conceal the correlation of input and output at the edges of the network using end-to-end correlation attacks. Note, however that protection of anonymity is gauranteed only within the borders of protected "area" and those "edges" are outside.

In general, traffic analysis is the strongest attack class (timing attacks, message volume attacks, intersection attacks) against which implemented anonymity networks are most vulnerable. We admit, that also our model described in this report is able to tackle this problem only partially.

Our input model is asynchronous. By an asynchronous model we loose the ability of handling some time-related characteristics within a message flow and between simultaneous inputs, e.g. the ability of modelling the on-line/off-line periods of the users (intersection attacks) or the time duration of a specific communication (timing attacks). An extension to our models could be the modeling also such time related traffic characteristics.

3. Our contribution

We propose a generic ideal model for anonymous communication together with a proof system within UC framework, based on BPW's BRSIM/UC approach. It is shown how this model relates to and extends earlier models:

Wikström [15] published the first UC result for the specific task of mix-nets for static adversary. In contrary, we propose a proof framework for tasks of anonymous communication, in general. In our approach a real protocol is transformed into a (ultimately runnable) symbolic protocol. It is more granular, goes down to arbitrary implementation, where the "building blocks" are the cryptographic primitives used in cryptographic protocols.

Hevia [10] introduced a definition of anonymity based on the notion of computational indistinguishability for global passive adversary. Our approach provides equivalently strong definition of anonymity with the significant advantage that our ideal system is also part of a proof system for assessment of anonymity provided by different realizations. We extend these results also to adaptive adversary.

We show a wireless example with a Group-based Private Authentication protocol [1].

4. The real system

When we use the simulatability paradigm in a security proof of a protocol we compare views in two related systems: the real and the ideal system. Simulatability

essentially means, whatever can happen to certain users in the real system it can also happen to the same users in the ideal system.

Fig.2. shows the block scheme of the real system. Components of the real system are user machine *H*, protocol machines $P_1,...,P_n$ corresponding to the honest participants, dedicated machines (servers) $g_1,...,g_w$ of the anonymizer network and adversary A_R. Here we allow the extreme case of $w=0$, which we consider equivalent to the case when there is no dedicated network within supporting the anonymizer (an example is given in Section 6.).

Machines, which are compromised before the run of the system are incorporated into the adversarial machine. User machine *H* models the surrounding protocol environment. The adversary has an auxiliary channel to user machine *H*, which models extra capabilities of the adversary: e.g. the adversary is allowed to initiate actions which are standardly allowed to machine *H*.

A run of the system is initiated by user machine *H* by sending input to the first protocol machine (sender), where the input is a flow of messages. User machine *H* and adversary A_R form the so called *environment* for the protocol running on protocol machines and the anonymizer network. Adversary A_R has no access to the communication between user machine *H* and honest protocol machines. Based on its view of the run the adversary tries to break the protection of anonymity. The view of the adversary consists of the view of all communication channels overheared by it, the auxiliary channel to user machine *H* and the information gathered from compromised elements of the system.

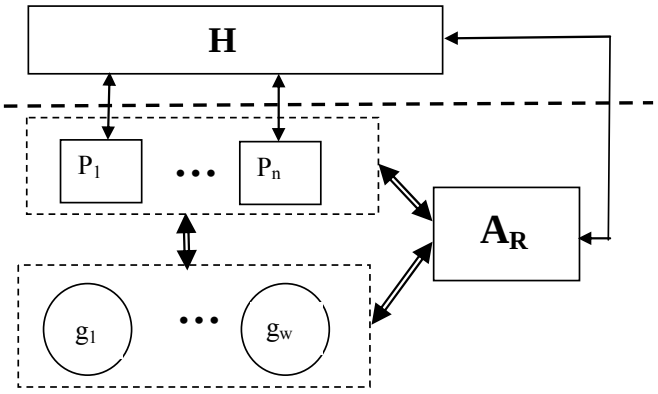

Fig.2. The real system

For simplicity of description we assume that all messages transmitted in the real and ideal systems alike have uniform length on all communication channels of the systems. Similarly, we assume that messages are not divided and composed: the whole message and only a single message is carried by a transmitted packet. Furthermore, we will call it as message only at the input/output of the anonymization system; any other bit strings transmitted within the system will be called packets (carrying messages).

Standard model of cryptography is assumed, i.e. a model of computation in which the adversary is only limited by the amount of time and computational power available. The adversarial machine is assumed to be PPT, where polynomial time is meant in a security parameter. (The concrete selection of the security parameter – typically a key length – depends on the actual cryptotechnology used in the implementation.) Below we do not assume trusted third parties. However, there may be applications, where allowing specific trusted third parties in modeling is not only unavoidable but also realistic (e.g. the public key infrastructure (PKI) model requires a trusted certificate authority). Fortunately, the UC approach favors hybrid models, where an UC-secure cooperating protocol (e.g. the protocol run by a trusted certificate authority) can be substituted by its abstract (ideal) model when analyzing the target protocol.

The adversary may launch the following attacks:

- *eavesdropping* of links
- *initialization of new runs* between pairs of protocol machines
- *access to current state or the entire view* of compromised machines
- *active attacks against runs* from compromised machines

Eavesdropping adversary can be global or local (partial). A *global adversary* watches all links. A *partial eavesdropping adversary* can only eavesdrop on part of the network. (A channel which corresponds to an eavesdropped link is called *insecure* or *"authenticated"* (read/write or read only, resp.) and a channel outside the accessibility of the attacker is called *secure*.)

A *static adversary* has to do corruption before the run, i.e. malicious machines are malicious from the beginning.
An *adaptive adversary* may do corruption during the protocol executions depending on the knowledge that he has already collected. Corrupted machines may be used by the adversary in a *passive* or an *active* way. In passive case corrupted machines follow the protocol, and the adversary has access to the current state or the entire view of the corrupted machine (*honest-but-curious* mode). In active case corrupted machines may deviate from their protocol arbitrarily (*Byzantine* mode).
In the model of the system with adaptive adversary we assume a special channel between the adversary and the protocol machines and servers. Such a channel is used by the adversary to control the compromised machine (send compromisation request, transmit state/view of the machine, load program in Byzantine mode etc.).

The dashed line (Fig.2.) indicates the interface, where the view of user machine *H* is defined. In case of anonymity protocols the aim of the attacker is to gain anonymity-related information and such an attack (typically) might not appear on the interface between the user machine and the honest protocol machines. What might indicate a successful attack could appear in the communication between the adversary and the user machive via the auxiliary channel (recall, user machine models the surrounding protocols). Note, this auxiliary channel is also included in the interface (dashed line in Fig. 2.). (1)

(1) Recall, any algorithm can be run by the adversary within computation constraint. Obviously, those algorithms are interesting for us among them, which can produce a view at the interface of the real system which can lead to non-negligibly distinct view.

Indistinguishability of the view by user machine has to be kept at this interface. This security requirement can equivalently be formulated in the following way: the adversary, according to the accessible information may have a guess on anonymity-related information. Accordingly, the adversary may send the corresponding guess to user machine *H*. Assume the user machine has a special output port to indicate if an adversarial guess is correct or not (note, machine *H* is able to decide it). If this guess is correct, machine *H* sends a corresponding output bit (1/0 – success/failure) to a specific output port.

Asynchronous system model is considered with the adversary as a master scheduler.

Asynchronism is assumed also in a different (but related) meaning: the input message flows are time processes. In the models presented in this report, we do not model the timing of messages within a flow except their time order and the exact timing between simultaneous flows.

5. The ideal system

When we define an ideal system we abstract away the implementation details of the concrete systems. An ideal functionality may be found to be unrealizable. In such a case it may be necessary to relax the functionality by leaking more information to the adversary. Additionally, in our approach the ideal functionality is not a "standalone" definition of ideality, but a part of a proof system, which defines a "specification".

For instance, if there is only a single active sender and the adversary is able to detect the activity of a potential sender then, obviously, no sender anonymity can be guaranteed. In particular, we should not require from a generic anonymizing system to ensure sender anonymity; such a property should be supported by the application environment. It is a standard assumption that a global passive adversary is aware of the sending protocol machines, while in case of wireless systems (e.g. wireless RFID/sensor) it is a plausible assumption the adversary cannot identify an active wireless sender by overhearing the wireless media shared by all participants of the system.

In case of anonymous communication it is especially important to define the set of *tolerable imperfections*, which are also built into the ideal system. This way we define the information, which could not be protected at an acceptable cost and leaked to the adversary. The set of tolerable imperfections highly depends on the actual application. Tolerable imperfections (ToI) have an important technical aspect in the simulatability approach. Even the ideal system shows ideal properties only up to these constraints.

Fig.3. shows the overview of the ideal system. Components of the ideal system are the trusted host F_{acom}, honest participants (protocol machines) $P_1,...,P_n$, adversary A_I and furthermore user machine *H*. Protocol machines corrupted by the adversary (before the run) are included into the adversary.

User machine *H* communicates with protocol machines: initializes a new run and receives the output of the run. In Fig.3. the dotted line delimits the *view* of user H, which must be indistinguishable in the real and ideal system.

Trusted host F_{acom}, running the ideal functionality is an important element of the system. Protocol machines (honest and adversarial) communicate with each other via F_{acom} by sending commands to it. Trusted host F_{acom} contains also a database, which stores the history of all operations made by protocol machines with cooperation of machine F_{acom}. F_{acom} is a set of number w connected machines, the corresponding communication graph G with nodes (servers) $g_1,...,g_w$.

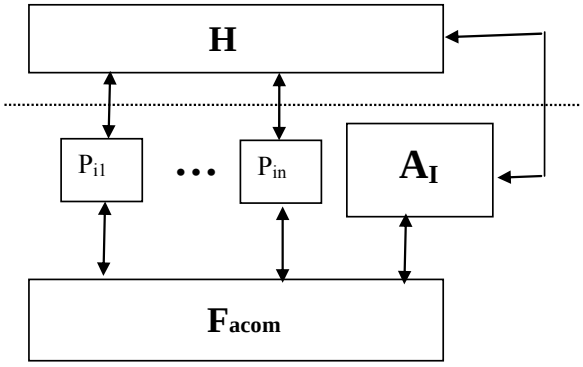

Fig.3. The ideal system

Cryptographic primitives (used by protocol machines and servers) are abstracted away. Cryptographic elements within the ideal system are ideal and based on composable cryptolibrary. The corresponding ideal functionalities are also part of trusted host F_{acom} (Fig.4.).

User machine *H* and adversary A_I is the *environment* for the (ideal) protocol running on protocol machines and trusted host F_{acom}. Adversary A_I does not see the communication between user machine *H* and the (honest) protocol machines, and similarly between trusted host F_{acom} and the protocol machines. In general, the adversary is allowed to watch any communication within the ideal system just via the mediation of the trusted host. The adversary based on its view of the run tries to break the protection of anonymity. The view of the adversary consists of the view of communication channels accessed via the trusted host, the auxiliary channel to user machine *H* and all the information gathered from/by corrupted elements.

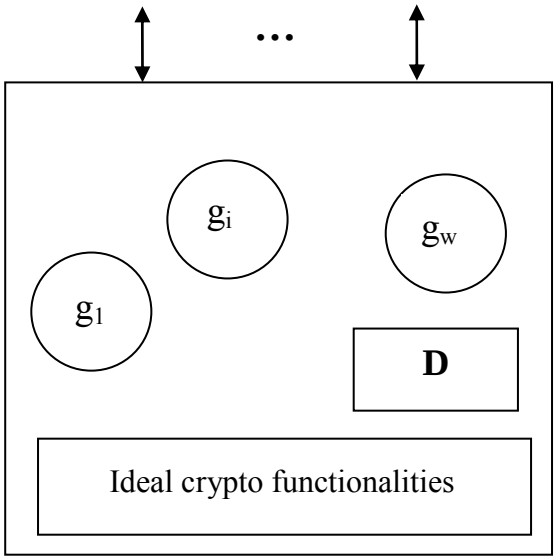

Fig.4: Trusted host F_{acom}

5.1. Running the ideal system

The main elements of the ideal system are user machine *H*, ideal host F_{acom} and adversary A_I. Adversary A_I plays also the role of a master scheduler. It schedules the user machine *H* and the ideal host F_{acom}. When the control is obtained by machines *H* or F_{acom}, they perform their next honest steps prescribed by their algorithms and give the control back to the adversary. When the control is at F_{acom}, it schedules the nodes within.
For easier following of the working of the ideal system, below we sketch the tasks of principals without rigorously emphasizing the role of the master scheduler. It is easy to produce such more formal version from the description below.

- Suppose *user machine H* initializes a run of the system by selecting the first sender-receiver pair of protocol machines (P_s, P_r) and a message m for transmission. Input (P_s, P_r, m) is forwarded to trusted host F_{acom} by protocol machine P_s ($Send(P_s, P_r, m)$ command). Trusted host stores data corresponding to communications into database *D*.

  Trusted host F_{acom} makes an entry for the input message in database *D* (see BPW's formalism e.g. in [2])

  $$D :\Leftarrow (ind := size++,\ type := data,\ \arg := (P_s, P_r, m),\ hnd_{P_s}) \qquad (1)$$

A participant is allowed to access an "information element" stored in the database of the ideal functionality iff he has a handle to it. E.g., a sender machine P_s gets a handle (hnd_{P_s}) to all fields of entry (1).

Trusted host F_{acom} informs the adversary about the reception of a new message and outputs a handle hnd_A for the adversary. The information to which this handle allows access depends on the set of tolerable imperfections. Having this handle the adversary, at minimum, has access only to the index and the type field. By this special handle we model that the adversary has no access to information in excess to the fact that a message has been sent to F_{acom} for anonymous transmission. However, for instance, according to the set of tolerable imperfections the adversary may know also the identity of the sender, when the handle allows the adversary to access also the corresponding element (P_s) of field *arg* within the entry.

- Suppose *adversary* decides to attack:
  - Adversary may compromise participants of the system (protocol machines and nodes within trusted host) by sending appropriate command to the actual input ports of the corrupted party.
  - Adversary may initiate message sending between any protocol machines via user machine *H*.
  - Adversary may send a guess on anonymity-related information to user host *H*.

- Suppose *trusted host* makes a step of anonymization. If this action is associated with a transmission of a packet within the trusted host and within the potential view of the adversary, then F_{acom} stores an entry into the database:

$$D :\Leftarrow (ind := size++,\ type := anon-data,\ \arg := (g_i, l_i, c_i),\ hnd_A) \qquad (2)$$

and sets a handle to the adversary to an abstract descriptor of the communication within F_{acom}. The content of field *arg* is (g_i, l_i, c_i), where l_i is a list of indices of messages carried by packets transmitted by node g_i, number c_i is the size of the list (equals to the number of transmitted packets).

F_{acom} informs the adversary about an anonymization step by sending handle hnd_A which allows adversary to access to the fields of entry (2) as follows: g_i: access depending on ToI, c_i: access, l_i: no access.

- Suppose *trusted host* F_{acom} decides to output: trusted host sends all output messages to all receivers. Adversary is informed about this transmission by obtaining handle to entry

$$D :\Leftarrow (ind := size++,\ type := output, \arg := (g_{i_1}, l_{i_1}, c_{i_1}, ..., g_{i_n}, l_{i_n}, c_{i_n}),\ hnd_A) \quad (3)$$

where the definition of the content of field *arg* and the access capabilities via handle hnd_A are as above.

The activated receiver P_r selects those messages, which are destined to it and forwards those to user machine *H*.

- Suppose *trusted host* F_{acom}, according to the level of its corruption, decides to give up any further anonymization effort and gives control to the adversary. Henceforth the ideal system no longer guarantees anything.

The aim with sending all output messages to all of the receivers by the trusted host is to prevent the adversary to link messages with receivers.

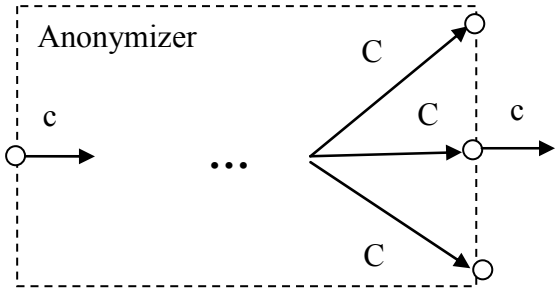

Fig.5. Illustration to receiver anonymity

Consider sending a single message (Fig.5). The existence of communication cannot be hidden from the adversary. Therefore the protection cannot be done simply by coding the packet. Receiver anonymity cannot be achieved otherwise just by sending the same packet to all receivers (packet *C* in Fig.5.). Because the adversary sees the bit sequence representing the packet, the identity of the receiver (in general, all anonymity related information not within ToI) should be encoded to protect against an overhearing adversary. Just the intended receiver should be able to decode the packet and select it from the stream of output packets sent together with packet *C* to all the receivers.

5.2. Comparison to earlier proposals

5.2.1. Wikström's ideal functionality

Our ideal system is not only an ideal functionality, but (in detailed form) it is a runnable system. It is a generic ideal system for anonymous communication with a general model for the adversary. We show that Wikström's ideal functionality F_{MN} is a specific version of F_{acom} under equivalent static adversaries.

Proposition 1: Ideal functionality F_{MN} is a specific version of generic ideal system F_{acom}.

Proof: The proof is based on the following observations:

Servers $g_1,...,g_w$ in F_{acom} correspond to mix-servers $M_1,...,M_k$ of F_{MN}, where $w = k$. Ideal functionality F_{MN} is defined for single message inputs, which is a a specific selection of inputs in ideal system F_{acom}.

In F_{MN} real messages are sent to the output (see in the Appendix). Considering message $M = (P_s, P_r, m)$ and assuming that payload *m* does not contain anonymity-related information, payload *m* may be used as a pointer to message *M* instead of an abstract substitute (as in the definition of F_{acom}) without sacrificing ideality of the system. (In model F_{acom}, we wanted to avoid including real payload into the output model which would tacitly assume some realization.)

In F_{MN} the adversary is aware of the identity of active transmitting elements (sending protocol maschines, servers), which assumption can be included in ToI in F_{acom}. Static adversary of F_{MN} is set also in F_{acom}.

Ideal system F_{acom} decides to send output according to the corresponding rule of F_{MN}.

■

5.2.2. Indistinguishability based anonymization vs. simulatability

The concept of indistinguishability-based ideal anonymizer with global passive adversary (henceforth called IND-anonymizer) has been introduced in [10]. Let *Q* be an anonymous communication scheme, similarly π_Q be the corresponding protocol in the real system.

In the indistinguishability game adversary A_Q selects a pair of inputs meeting ToI-constraints, which means the adversary interacting with the system cannot distinguish this pair purely from information leaked to it via ToI. Adversary A_Q sends the pair to the oracle of the indistinguishability game, which randomly selects one of them and initiates a run of the anonymizer with this message. Adversary A_Q as a global passive adversary is allowed to see communications during the run. Finally, adversary A_Q has to decide which message has been selected by the oracle.

Below we examine the relationship between indistinguishability based and our simulation based approach.

Recall, in the approach of simulatability the security of a realization means the following: for any adversary A_R against the real system there exists an adversary A_I attacking the ideal system such that the views of user machine *H* are indistinguishable. Black box simulation is a useful technique in an effort to prove

indistinguishability of the two systems (i.e. real and ideal). We define a simulator with a task to provide a real-like environment indistinguishable for adversary A_R, and to translate messages from symbolic to real and vice versa sent between trusted host F_{acom} and adversary A_R. Fig.6. shows the ideal* system I^* trusted host F_{acom} is extended with a universal simulator, SIM.
(Recall, when we use the ideal model as a component within a (hybrid) protocol, we use just the abstract functionality F_{acom}, i.e. the simulator is used only in analyis if an implementation UC-realizes ideal functionality F_{acom}. Sometimes the simulator is considered as part of the adversary, however, in the proof below it is more convenient to consider it as an extension to the ideal system.)

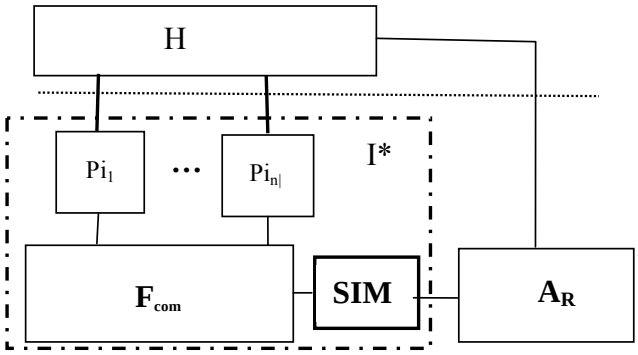

Fig. 6. Ideal* system with simulator

The state of machine SIM consists of a database D_s and the current handle for the adversary $curhnd_a$. Database D_s stores the already mapped adversarial handles. Each entry in D_s has attributes: $(hnd_a, word, add\_arg)$.

SIM receives an input from F_{acom}:
Each symbolic element δ in the ideal system, which is associated with a real quantity d in the real system and which is accessible by the adversary in the real system, is translated into a real element (bitstring) d^* meeting the following requirements:

1. d^* provides no information about d for the adversary (in excess to a priori)
2. d^* and d are indistinguishable variables by the adversary
3. generation of d^* is polynomially constrained

d^* is stored in D_s as a new *word*. For brevity, we call d^* as dummy (dummy substitution).

An adversary sees only the messages produced by SIM, while the correct messages will be delivered to the receiving protocol machines. Informally, the adversary gets no information in excess to the information known a priori (ToI included).

SIM receives an input from the adversary:
SIM parses the real input, builds up the corresponding symbolic terms and stores them in D_s.

First, consider a global passive adversary, which in the next step will be generalized to an adaptive adversary. The proof technique of Proposition 2. below is adapted from a proof of Canetti [6].

Proposition 2: Under global passive adversary an anonymous communication scheme *Q* is an IND_anonymizer if and only if π_Q UC-realizes ideal functionality F_{acom}.

Proof:

UC → IND

We show that if *Q* is not an IND_anonymizer then π_Q fails to securely realize ideal functionality F_{acom}. Let A_Q denote an adversary which can successfully attack scheme *Q*, i.e. A_Q is able to guess random bit *b* with probability $1/2+\varepsilon$ in the ToI-constrained indistinguishability game, where probability ε is non-negligible.

We construct an environment $Z=(H,A)$, which can distinguish the real and the ideal* systems with the same (ε) advantage. Adversary *A* chooses test messages and transmits those through the anonymizer via *H* and finally *H* decides according to the output (from the receivers and adversary *A*).

Adversary *A* runs adversary A_Q in a simulation of the IND_anonymizer game: A_Q generates a pair of ToI-constrained test messages (m_0,m_1); *A* chooses random bit *b* and sends message m_b to the system (real or ideal*) via *H* for anonymous transmission; adversary *A* lets adversary A_Q see the same view of the system; when A_Q outputs decision b', *A* outputs $b\oplus b'$ via *H*. (Here message means an input for a run of the system, which actually means a set of input message streams.)

When environment *Z* has access to the real system, then *A* outputs 0 with probability $1/2+\varepsilon$, but when it accesses the ideal system this probability becomes exactly $1/2$.

Note, if the pair of test messages, would not meet ToI-constraint, then adversary *Q* could distinguish the pair with success probability 1 either it sees the view of the real or of the ideal system, consequently adversary *A* could not distinguish the two worlds by running adversary A_Q.

IND → UC:

We show that if there exists an environment *Z*, which can distinguish the real and the ideal* systems, where protocol π_Q runs in the real system, then scheme *Q* is not an IND_anonymizer.

Assume environment *Z* is able to distinguish the real and the ideal system with advantage ε, then we show that there exists an adversary A_Q which can successfully attack scheme *Q*, by being able to guess random bit *b* with probability $1/2+\varepsilon/(2n)$ in the indistinguishability game with ToI-constraint, where *n* is the number of test messages sent by *Z* to the tested system. The proof uses standard hybrid proof technique:

- A_Q randomly chooses an element *h* from set $\{1,...,n\}$;
- in step *i*, $1 \le i < h$ environment *Z* wants to transmits a message m_i; adversary A_Q transmits message m_i through its own tested system, where the view of the run is accessible also for environment *Z*;
- in step *h*, when environment *Z* wants to transmit message m_h, adversary A_Q forms a pair (m_h, m^*) of messages, where m^* is a random message with ToI-constraint; pair (m_h, m^*) is sent to the IND-anonymizer oracle, which randomly chooses one of them; adversary A_Q transmits the chosen message through its own tested system, where the view of the run is accessible also for environment *Z*;
- in step *i*, $h < i \le n$, when environment *Z* wants to transmits a message m_i; adversary A_Q transmits random message m_i^* through its test system; the view of the run is accessible to environment *Z*;

Note, if an independent random substitute is transmitted instead of a real message, the chance of success for the adversary to gain anonymity-related information from viewing the run of the anonymizer is equivalent to viewing the run of an ideal* system:

Note, in a random input m_i^* both the sender/receiver pairs and the payload are chosen at random. Therefore, if such an input is given to the (real) test system of adversary A_Q instead of message m_i intended by adversary *A*, then environment *Z* sees no more m_i-related anonymity information ("who sends to whom") than by seeing the view of the ideal* system driven by input m_i.

It follows, the two extreme hybrids is equivalent to accessing a real system and an ideal* system, respectively.

The advantage in indistinguishability game is probability $1/2+\varepsilon/(2n)$, which follows from arguments of standard hybrid technique. ■

Corollary 1 of Proposition 2: An UC-realized ideal functionality F_{MN} meets the indistinguishability requirement.
Proof: Straightforward consequence of Proposition 1 and 2.

5.2.3. The case of adaptive adversary

Note, that in the proof of Proposition 2, adversaries *A* and A_Q use each other as a black box, simulating inputs and taking outputs/views. At first glance, such a black box technique does not seem to be extendable to active/adaptive adversaries. As it was explained briefly in Section 4, an adaptive adversary has special communication channels to machines for corruption purposes.

If the adversary uses a corrupted machine in passive way, the view of the adversary is extended with the information transmitted in the special corruption channel. For instance, if an adaptive adversary *C* uses another adaptive adversary *B* as black box, then adversary *C* is able to simulate a corruption attack for itself by having access to the extended view of adversary *B*. The extension of Proposition 2 is straightforward to such adaptive/passive corruption adversary.

Now consider the case when the corrupted machine is used in Byzantine mode. If we can assume that adversary *C* is able to follow the behaviour of the corrupted machine by having access to the communication between adversary *B* and the corrupted machine, then we can extend Proposition 2 also to an adaptive/active corruption adversary. We briefly consider two plausible scenarios to illustrate the viability of such an assumption:

By definition, adversaries have perfect knowledge about all HW/SW details of an honest machine. An adversary may use the implemented SW without changing any part of it, such that the adversary runs these SW components in a way different from the honest mode. The calls of adversary B to these SW components is followed also by adversary C, therefore the latter adversary keeps maintaining perfect knowledge about the operation of the corrupted machine.

In the case when a new program code is downloaded to the corrupted machine, adversary *C* also sees the code. We assume that adversary *C* is able to analyze the new code perfectly. If, as assumed, the HW is intact, adversary *C* is also aware of the functioning of the fabricated machine.

The proof of Proposition 2 can be extended to adaptive adversary with the following additions. Note furthermore that adversary B's and C's "own" real systems are not different in the sense that being in equal state they respond the same way to inputs. For instance, if adversary *B* simulates the environment for adversary *C* by channeling all communications of *C* to its own (i.e. *B*'s) real system, adversary C will not recognize any difference. In particular, this is true also for corruption messages via corruption channels. When an ideal* system is accessed this way by adversary *C*, the simulator (SIM) will translate between the real and symbolic world, in particular also for corruption aimed communication.

Under these assumptions:

Corollary 2 of Proposition 2: Assume an adaptive adversary, the corruption operation of which can be simulated by overhearing the communication channels used for controlling the corruption attack. An anonymous communication scheme Q is an IND_anonymizer with such an adaptive adversary if and only if π_Q UC-realizes ideal functionality F_{acom} with corresponding adaptive adversary.

6. Examples

A wireless application: private authentication

In private authentication two principals require that they reveal their identities to each other, however they wish to reveal nothing to others. Common authentication protocols cannot provide an immediate solution, because these protocols often send names and credentials in cleartext, allowing any eavesdropper to see them.

Sender anonymity of wireless client-server application (RFID, wireless sensors) is considered. Wireless clients want to authenticate to a wireless server and an adversary overhears the wireless channel. In contrast to wired networks (Internet) the adversary, in general, is not able to identify the active wireless transmitter from the reception of radio signals. An active Dolev-Yao style adversary is assumed, which is able to modify the transmitted wireless packets, it controls the scheduling of the output, the concurrency and the interaction between subsessions.

Protocol 1:

First we consider the following protocol with a single message sent by client P_i to server S:

$$P_i \to S: \; E_k(P_i) \tag{4}$$

where E_k is a (probabilistic) symmetric key encryption with predistributed secret key k shared between client P_i and server S. The server stores the secret keys of all clients. It searches through this set of keys until it can decode a valid identifier. This searching complexity is decreased by the following Group-based Private Authentication protocol [1].

Protocol 2:

$$S \to P_i: \; R_1$$
$$P_i \to S: \; E_{k_g}(R_1, R_2, P_i), E_{k_c}(R_1, R_2)$$

where R_1 and R_2 are one time random values chosen by the server and the client resp., k_g and k_c is the group key and the client key of client P_i.
The server challenges the client, which responds with a message consisting of two ciphertexts. The server knows all group keys and client keys. First it tries different group keys in a fixed order until it can decrypt a plaintext with challenge R_1 in the first field. Next the server searches through client keys within the identified group until it decrypts a plaintext with the challenge in the first field.

It is assumed that the secret keys have safely been distributed before running the system.

Trusted host F_{acom} can be adapted (simplified) to this application: There is no inner network within the trusted host ($w = 0$). There is an ideal functionality for symmetric key encryption within host F_{acom}.

The simulator in the ideal* system has to produce a ciphertext which leaks no secret information and indistinguishable from the real one. In other words, the simulator has to construct an indistinguishable ciphertext knowing neither the secret key nor the actual identifier (plaintext). For dummy plaintext it uses a constant identifier, assuming the encryption guarantees indistinguishability in plaintext. The missing key is simulated by choosing a randomly selected secret key.

For instance in case of Protocol 1 the following entry is set in database D:

$$D :\Leftarrow (ind := size++,\ type := enc,\ \arg := (E_{k^*}(P^*)),\ hnd_{P_i})$$

where P^* is a dummy (constant) identifier and k^* a one-time random key.

On adaptive adversary in mixing networks

It is known, that if the adversary is active (i.e., Byzantine) then adaptive security is strictly stronger than non-adaptive security, regardless of the values of all other parameters. Here we consider an example, which demonstrates similar advantage even in case of passive adaptive adversary.

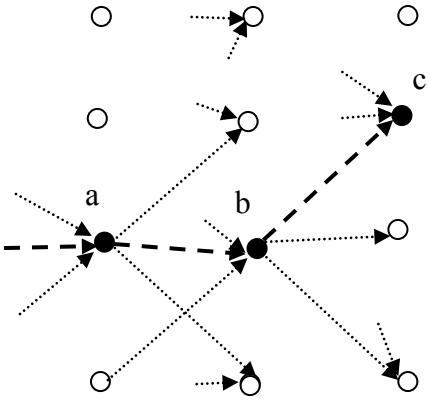

Fig.7. An example message transmission graph of a mixing network

As any global passive adversary the adaptive adversary sees the built up of the message transmission graph as the messages proceed from the senders to the receivers (Fig.7.). Transmitting nodes choose the next receiving nodes by some randomized algorithm. When the adversary corrupts a node the relationship of edges entering and leaving the node becomes known to the adversary.

The task of guessing the receiver at the output of the anonymizer for an input message is equivalent of guessing the corresponding route.

First consider the task of exploration of the route for a given input message (see dashed line in Fig.7). Obviously, if the adaptive adversary is allowed to decide about corruptions at the end of the run, he will corrupt nodes *a,b* and *c* in order.

A more general task is when the aim of the adversary is to increase the success of guessing as many paths from transmitters to receivers as he can under the constraint of limited number of corrupted nodes and the feasibility of computations. A corresponding question is the optimal corruption algorithm, its complexity together with its advantage over a blind selection nodes even in the special case of a matrix of nodes where the left/right column of nodes are the senders/receivers resp. and a message proceeds in the matrix by selecting the next node uniformly random.

For instance, a feasible ad hoc algorithm of choosing a node with highest degree for corruption (in and out degrees of a node are equal) in each column guessed to have strict advantage against a random selection.

7. Conclusion

With this report we tried to turn the attention to the BRSIM/UC proof system of BPW in security evaluation of protocols for anonymous communication. Up to our knowledge this is the first publication which provides a generic model for anonymous communication together with a proof system. Despite of generality of the model, it does not cover all of the rich aspects of anonymous communication having impact on security. Time related (and attack sensitive) characteristics of input flows seems an interesting issue for an extension of models drafted in this report.

References

[1] G. Avoine, L. Buttyán, T. Holczer and I. Vajda. Group-Based Private Authentication. *In Proceedings of the International Workshop on Trust, Security, and Privacy for Ubiquitous Computing* (TSPUC 2007), IEEE, 2007.

[2] M. Backes, B. Pfitzmann, and M. Waidner. A universally composable cryptographic library. *IACR Cryptology ePrint Archive*, Report 2003/015, http://eprint.iacr.org/, January 2003.

[3] L. Buttyán, T. Holczer and I. Vajda. On the Effectiveness of Changing Pseudonyms to Provide Location Privacy in VANETs. *In Proceedings of the Fourth European Workshop on Security and Privacy in Ad hoc and Sensor Networks* (ESAS2007), Springer, 2007.

[4] L. Buttyán, T. Holczer and I. Vajda. Optimal Key-Trees for Tree-Based Private Authentication. *In Proceedings of the International Workshop on Privacy Enhancing Technologies* (PET), June, 2006, Springer.

[5] L. Buttyán, T. Holczer and I. Vajda. Providing Location Privacy in Automated Fare Collection Systems. *In Proceedings of the 15th IST Mobile and Wireless Communication Summit*, Mykonos, Greece, June, 2006.

[6] R. Canetti. Universally Composable Security: A New Paradigm for Cryptographic Protocols". *Cryptology ePrint Archive: Report 2000/067.* (received 22 Dec 2000, revised 13 Dec 2005)

[7] D. L. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2): pp.84-88, 1981.

[8] G. Danezis, C. Diaz, and P. Syverson. Systems for Anonymous Communication. *In CRC Handbook of Financial Cryptography and Security, CRC Cryptography and Network Security Series,* B. Rosenberg, and D. Stinson (Eds.), Chapman & Hall, pp. 341-390, 2010.

[9] F.D. Garcia, I. Hasuo, W. Pieters, and P. van Rossum. Provable anonymity. In *Proc. of the 3rd ACM Workshop on Formal Methods in Security Engineering – FMSE'05*, pages 63–72. ACM Press, 2005.

[10] A. Hevia, D. Micciancio. An indistinguishability-based characterization of anonymous channels. In Nikita Borisov and Ian Goldberg, editors, *Privacy Enhancing Technologies: Eighth International Symposium, PETS 2008*, pages 24-43. Springer-Verlag, LNCS 5134, July 2008.

[11] T. van Le, M. Burmester and B. de Medeiros. Universally Composable and Forward Secure RFID Authentication and Key exchange. *ACM Symposium on Information, Computer and Communications Security, ASIACCS 2007*, Singapore, March 2007

[12] A. Pfitzmann and M. Köhntopp. Anonymity, unobservability, and pseudonymity - a proposal for terminology. In Hannes Federrath, editor, Designing Privacy

Enhancing Technologies: *International Workshop on Design Issues in Anonymity and Unobservability*, pp. 1-9. Springer-Verlag, LNCS 2009, July 2000.

[13] B.Pfitzmann, M. Waidner. Composition and Integrity Preservation os Secure Reactive Systems. *7-th Conference on Computer and Communication Security of the ACM*, pp.245-254, 2000.

[14] M. Reiter and A. Rubin. Crowds: Anonymity for web transactions. *ACM Transactions on Information and System Security*, 1(1): pp.66-92, 1998.

[15] D. Wikström. A universally composable mix-net. In *Theory of Cryptography TCC'04*, volume 2951 *LNCS*, pages 317–335. Springer-Verlag, 2004.

[16] I.Vajda. Cryptographically Sound Security Proof for On-Demand Source Routing Protocol EndairA. *Cryptology ePrint Archive Report 2011/103*. http://eprint.iacr.org/2011/103.pdf

[17] I.Vajda. Framework for Security Proofs for Reactive Routing Protocols in Multi-Hop Wireless Networks. *Cryptology ePrint Archive Report 2011/237*. http://eprint.iacr.org/2011/237.pdf

[18] I.Vajda. New look at impossibility result on Dolev-Yao models with hashes. *Cryptology ePrint Archive Report 2011/335*. http://eprint.iacr.org/2011/335.pdf

[19] I.Vajda. Non-malleable public key encryption in BRSIM/UC. *Cryptology ePrint Archive Report 2011/470*. http://eprint.iacr.org/2011/470.pdf

Appendix

Ideal functionality F_{MN} (Mix-net) [15]:

1. Initialize list $L = 0$, and set $J_P = 0$ and $J_M = 0$.
2. Suppose $(P_i, Send, m_i)$, $m_i \in G_q$ is received from C_I. If $i \notin J_P$, set $J_P \leftarrow J_P \cup \{i\}$, and append m_i to the list L. Then hand $(A, P_i, Send)$ to C_I.
3. Suppose (M_j, Run) is received from C_I. Set $J_M \leftarrow J_M \cup \{i\}$. If $| J_M | \geq k/2$, then sort the list L lexicographically to form a list L' and hand $\left((A, M_j, Output, L'), \{(M_l, Output, L')\}_{l=1}^{k}\right)$ to C_I. Otherwise hand C_I the list (A, M_j, Run).