MIT/LCS/TM-250

ON CONCURRENT IDENTIFICATION PROTOCOLS

ODED GOLDREICH

December 1983

On Concurrent Identification Protocols

Oded Goldreich
Laboratory for Computer Science
MIT,room NE43-836,Cambridge,Ma 02139

Abstract

We consider communication networks in which it is not possible to identify the source of a message which is broadcasted through the network. A natural question is whether it is possible for two users to identify each other concurrently, through a secure two-party protocol. We show that more than the existence of a secure Public Key Cryptosystem should be assumed in order to present a secure protocol for concurrent identification. We present two concurrent identification protocols: The first one relies on the existence of a center who has distributed "identification tags" to the users; while the second protocol relies on the distribution of "experimental sequences" by instances of a pre-protocol which have taken place between every two users.

Keywords: Cryptographic Protocols, Concurrent Identification, Public Key Cryptosystem Applications.

This research was carried out at the Computer Science Department
Technion - Israel Institute of Technology

1. Introduction

Let N be a set of users in a communication network in which **it is not possible to identify the source of a message broadcasted on the network** . Thus, identification of the source of a message can only rely on the content of the message. Clearly, this would require some sort of a secure authentication scheme as well as a secure protocol which makes use of it.

The task of reaching concurrent identification is somewhat more involved. It requires not only that identification takes place but also that it takes place concurrently; i.e that through this process there would be no situation in which one party had a "substantial" advantage in guessing and/or computing his counterpart's identity. Methods for reaching concurrent identification may be of value in certain business environments in which transactions are carried out in two stages: first reaching an anonymous agreement and only then yielding the identities of the parties to the agreement, as quickly as possible.(An example of such an environment is a future stock exchange without brokers[dealers] or even a present stock exchange controlled by an agency that wishes to prevent biased deals.)

Clearly, if one allows the participation of trusted third parties in the concurrent identification process, trivial solutions exist. However,we are interested in the existence of two-party protocols through which concurrent identification takes place (hereafter referred to as *Concurrent Identification Protocol*s or as *cip*s).

In Sec.2 we show that the mere existence of a PKCS (Public Key Cryptosystem [DH]) and a public file of all public keys does not suffice for the existence of a secure cip in the net (i.e. there exists no secure cip in such a net). It is also proven that if users are not supposed to eavesdrop on all instances of a cip (or inform all users on such instances) then a secure cip can not be deterministic.

In Sec.3 we present a cip which relies on a trusted center which has prepared and distributed "identification tags" to the users at the time the net has been established. (**This center does not participate in the cip**!) The number of transmissions needed to distribute these tags is linear in the number of users; thus the complexity of establishing a net in which this cip can be used securely is still linear in the number of its users. This fact combined with the simplicity of the cip itself makes its implementation reasonably practical.

In Sec. 4 we present a secure cip which does not rely on the honesty of some center nor even on its mere existence. Instead this cip relies on information which has been passed between every pair of users , via instances of a pre-protocol which have taken place at the time the net was established. The fact that the pre-protocol is fairly complicated combined with the fact that $O(|N|^2)$ instances must take place, cause this concurrent identification scheme to be impractical ,especially for large networks. However it demonstrates that concurrent identification can take place even if no center exist (at the time the net has been established as well as later).

In both Sec. 3 and 4 we assume the existence of secure cryptosystems, in particular the existence of a secure public key cryptosystem (PKCS)[DH]. We (also)

assume that the instances of these cryptosystems are free of any relation other than the cancellation of encryption by the corresponding decryption and vice versa. Note that this assumption yields that it is impossible to get "valuable" information about a plaintext without being able to get it as a whole. We consider the freeness assumption to be a natural one. We use it in the implementation of the OT subprotocol (see Appendix A). (The role of the OT is hereafter discussed.)

A natural problem which arises when designing identification protocols is the *replay problem*, which is hereafter described. User A may try to impersonate user B by using information B has revealed to him in previous instances of the identification protocol. Note that this information has been used to authenticate B and can be used by A to cheat C, unless the protocol has features which prevent such an attempt to cheat. The solution, as proven in Sec. 2 must contain randomization of some kind. In case of simple identification it is enough to ask for a signature to some time dependent message. (Note that this can not be done trivially in a cip since a signature to any message will immediately reveal the identity of the signer.)

To solve the replay problem in the concurrent identification protocols presented in this paper we use an *Oblivious Transfer* (OT) subprotocol. The notion of OT was first introduced and implemented by Rabin [R]. Another definition of OT, which we believe to be more natural, was suggested by Even,Goldreich and Lempel [EGL] (and implemented using any PKCS deemed secure). By their definition an OT of a recognizable message ,M, is a protocol by which a *sender* ,S, transfers to a *receiver* ,R, the message M so that R gets M with probability one half while for S the a-posteriori probability that R got M remains one half. In this work, however, we refer to OT as to a protocol which allows S to transfer a set ,$\{ M_i \}_{i=1}^{t}$, of recognizable messages so that R gets exactly one message. R gets M_i with probability p_i while for S the a-posteriori probability that R got M_i remains p_i.(The p_i's are fixed possitive reals which sum up to one.)

2. Necessary Conditions for the Existence of a CIP

It was already mentioned that no cip (as well as no identification protocol) can exist in a net if it is not assumed that the users are provided with some secure cryptographic identification scheme. We will assume the existence of both a secure conventional cryptosystem (e.g. the DES[NBS]) and a secure PKCS. However, we shall show that this assumption does not suffice to allow the existence of a secure cip, namely:

Theorem 1: A cip, which relies only on the existence of secure cryptosystems (the instances of which are free of any relation other than the cancellation of encryption by the corresponding decryption and vice versa) and a public file of all public keys , can not be secure.

proof: Since the cryptosystems are free of any relation other than the cancellation of encryption by the corresponding decryption and vice versa, it is

not possible to distinguish between instances of a cryptosystem without being able to identify at least one of them. Note that the only way a user can be identified is through his use of his secret decryption algorithm which can be verified by his publicly known encryption algorithm. (Note that no other way exists since the user's public encryption key is the only information known a-priori to his counterpart in the cip.) Let P be a cip. A user ,S, can cheat in the execution of P by using instances of cryptosystems he has secretly created (i.e. S secretly generates a new instance of the PKCS but does not make the encryption key public; when playing P he uses this secretly generated instance instead of his original instance). Note that S's counterpart can not tell whether S is cheating since he can not distinguish the case in which S uses his own instance from the case he is using a secretly generated one. Thus, P is not secure.

Q.E.D.

To conclude this section we show that a secure and deterministic cip can only exist under irreasonable assumptions, namely:

(i) Each user eavesdrops all instances of the cip and records the information he reads.

or

(ii) Each user notifies all other user about every instance of the cip he participates in.

Theorem 2: If neither (i) nor (ii) hold then a secure cip can not be deterministic.

proof We remind the reader that no active third parties exist. Let P be a deterministic and secure cip. Note that messages transmitted in P can not contain unforgable time dependent parts. (This is due to the fact that on one hand only the sender can authenticate the time the message was transmitted while on the other hand this authentication would yield his identity.) Consider an instance of P in which an honest user,A, plays the role of the first party and a dishonest user ,S, plays the role of the second user. To cheat A, S initiates P with another user B, such that S plays the role of the first party in this instance. S executes both instances simultaneously (i.e. transmits to A the messages he gets from B and vice versa). Since S is deterministic neither A nor B can differ, by looking at the messages they have receive, this case from the case P is really played by A and B. Since $A[B]$ does not eavesdrop the communication between S and $B[A]$, and $B[A]$ does not notify $A[B]$ on them, $A[B]$ has no alternative way to differ these cases. Thus, S will succeed in identifying A without disclosing his own identity , in contradiction to our assumption that P is secure.

Q.E.D.

3. A CIP which Relies on Preparations by a Trusted Center

*I*n this section we show how identification tags distributed, to the users, by a trusted center can grant the existence of a cip. The center can distribute these tags

at the time the network is established. The center must be trusted not to collaborate with any user, in the process of distributing the tags as well as during the time the cip is run. It is preferred that the center would seize to exist after distributing the tags.The tags will bear the center's signature and thus be unforgable. Every user can protect himself against the replay of his tags (by other users), by using a tag only once. Thus, the center should provide each user with enough tags.

We assume the existence of a secure PKCS (e.g. the RSA[RSA]) and of a conventional cryptosystem (e.g. the DES[NBS]).We also assume that all users have equel computing power.

3.1. The Identification Tag

Before describing the structure of the identification tag let us introduce some notation:

(i) F denotes a conventional cryptosystem and $F_K(M)[F_K^{-1}(M)]$ denotes the encryption[decryption] of M by F using the key K.

(ii) E_X , D_X will denote the encryption and decryption algorithms of user X (i.e. the PKCS's instance generated by X). Note that $D_X(M)$ can serve as X's signature to M.

(iii) C denotes the center.

(iv) N_X denotes the binary representation of X's name.

An *Identification Tag* (IT) of user X consists of three parts:

(1) The *header* , which contains an (unforgeable) encryption of X's name : $D_C(z, F_y(S), F_y(N_X))$,where y is a randomly chosen key (of length k) to F and z is a random "serial" number.

(2) The *anti-replay* part , which consist of n pairs of recognizable (and unforgeable) messages. The i-th pair denoted AR_i is $(D_C(z, L_i), D_C(z, R_i))$.

(3) The *certified key-bits* part , which consists of the bits of the key , which was used for the encryption of X's name, certified by the center: the certification of y_i (the i-th bit of y) is $D_C(z, i, y_i)$.

Note that all parts of a IT bear the same serial number and that they are signed by the center. User X is called the *legitimate holder* (or just the holder) of the above identification tag. (Note the although other users can have parts of X's tag only X can have all of it if he follows the cip described below properly.)

Remark: S , the L_i's and the R_i's are arbitrary , fixed messages (i.e. invariant of X ,y and z).

We remind the reader that these IT's will be distributed to the users by C at the time the network is established. Note that at that time only X has X's ITs. In the next subsection we will present a cip in which X uses one of his ITs to identify himself without yielding the entire IT. It will be shown that this prevents the replay of this IT by another user.

3.2. The Protocol

The cip described below uses an OT subprotocol which allows a user to send two recognizable messages such that : (1) his counterpart receives exactly one of them; (2) with probability one half the receiver receives the first message; (3) for the sender the a-posteriori probability that the first message was received remains one half; (4) if the sender tries to cheat the receiver will detect it with probability at least one half.

(An implementation of this OT is described in Appendix A and is based on ideas which first appeared in Even,Goldreich and Lempel [EGL].)

The cip proceeds as follows:

(The parties to the protocol are denoted A and B)
step 1:
  A chooses one of his unused ITs (hereafter denoted t_A)
  marks t_A as "used"
  and transmits t_A's header to B.
  B acts symmetrically transmitting t_B's header to A.
  (Each checks whether the center's signature
   to the header is authentic.)
step 2:
  for i=1 to n do begin
    A sends to B one element out of t_A's AR_i , via OT.
    B acts symmetrically w.r.t. t_B .
    (Each uses the cheat detection mechanism of the OT.)
  end
step 3:
  for i=1 to k do begin
    A transmits to B the i-th certified key-bit of t_A .
    B acts symmetrically w.r.t. t_B .
    (Each checks the signature certifying the bit received)
  end

3.3. Analysis of the Protocol and the Structure of the IT

Remarks (for $X \in \{A, B\}$)

(R1) The header of t_X establishes a linkage among X's name (although encrypted) the key y (which is used for the encryption of both N_X and the standard message S) and z (which is used as a serial number). It also provides information for the computation of y although this computation becomes feasible only during step(3).

(R2) The anti-replay of t_X allows X to protect himself against the replay of t_X. Note that if X uses t_X only in one instance of the protocol and execute this instance properly then he is (still) the only user in the net who knows both elements of each AR_i in t_X. (Note that his counterpart to the cip instance only

got one element out of each AR_i.) User Y ,$Y \neq X$, will succeed in replaying t_X only if he is asked in the OT of each AR_i (which occurs in step (2) of the protocol) to disclose the element of AR_i which is known to him. Note that for Y, both the element he is asked to disclose and the element known to him are randomly chosen out of an AR_i of t_X (this is due to the use of the OT in step(2)). Thus, the probability that Y will succeed in replaying t_X is bounded from above by 2^{-n}. Thus, a proper execution of step (2) of the protocol (only) assures the parties that the identification tags are in the hands of their legitimate holders.

(R3) The third part of t_X (which is exchanged in step (3) of the protocol) allows the gradual decrease in the time of computation which is required to extract N_X from the header of t_X. N_X is extracted by first finding the key y which transforms the message S into the cryptogram $F_y(S)$. Note that this computation becomes feasible (during step(3) of the protocol) only after the tag holder has proven himself to be the legitimate one (by succeeding in an unfaulty execution of step(2) of the protocol).

(R4) If the rate ,in which the time which is required to compute N_X given the header of t_X decreases, is considered to be too fast one may slow it down by using simple "exchange of half bit" schemes (e.g. Tedricks' schemes[T]).

(R5) The interleaving in step(2) of the protocol is not material.

We claim that this cip is secure provided the following assumptions hold:

(A1) A trusted center has distributed the identification tags described in sec. 3.1 to the legitimate holders.(The center is trusted not to convey any information about the tags he has provided user X to any other user.He is also trusted not to yield his signature algorithm.)

(A2) All parties have equal computing power.

(A3) Both the conventional cryptosystem and the PKCS used by the protocol are secure. (No one can forge C's signature. Extracting M from $F_K(M)$ given S,$F_K(S)$ and some of K's bits requires exhaustive search on all keys which match the known bits of K ; when no bit of K is known this computation is infeasible.)

Theorem 3: If the above assumptions hold and a user ,U, plays the protocol properly then the following hold:

(1) In any phase during the execution of the protocol,if U's counterpart can find out U's identity using expected time t then U can find out what is claimed to be his counterpart's identity in about the same expected time.
(2) If U's counterparts is honest U will find out his identity.
(3) If U's counterpart is impersonating then with high probability $(1 - 2^{-n})$ U will find this out before reaching a stage in which the computation of his identity is feasible.

proof: (1) follows from (R1),(R3),(A1),(A2) and (A3). (2) follows from (R1),(R3) and (A1). (3) follows from (R2) and (A1).

Q.E.D.

4. A CIP which Relies on Preparations by Instances of a Pre-Protocol

*I*n this section we (only) assume the existence of a secure PKCS. We show how a pre-protocol ,played between every pair of users,can grant the existence of a cip in the net. Note that we do not assume that there exists some (trusted) center and that we do not assume that all parties have equal computing power. (It should be stressed that we **do not** refer to the public file of the users' encryption keys as a center.) Since instances of the pre-protocol must take place between every pair of users, the result of this section ,although being of theoretical interest, is practical only for "small" networks. The purpose of the pre-protocol is to distribute *secure experimental sequences* which will be used in the identification process. These sequences will be unforgeable and will yield the identity of their legitimate holder[1] if some parts of them are read completely. However it will be possible to give away only small (still unforgeable) fragments of the sequence yielding only a "small amount of information" about their legitimate holder.

The idea behind the implementation of these experimental sequences is to allow a user to conduct experiments on the bits of another user's name. The experiment is gauranteed to give a result equal to the tested bit with some fixed probability greater than one half. Thus conducting enough experiments on a bit gives certainty of knowing its right value ; whereas on the other hand a single experiment does not give much information about the corresponding bit. The cip consist of letting each user experiment on each of his counterpart's name bits by just sending one entry in the experimental sequence. The implementation of a process which constructs experimental sequences will be discussed in subsection 4.4 . (Its essence is that this sequence will be built anonymously by the user who will later experiment on it. The sequence will be built by flipping a biased coin so that its builder will only know the expected value of an entry in it and not the concrete value. This will be achieved by using an OT.)

The idea of using a biased coin as a tool for exchanging a bit of information was suggested ,independently, by Lubi,Micali and Rackoff in their MiRackoLus paper [LMR]. It should be stressed that the problem they were facing was much more difficult and their solution (a coin the bias of which is determined by the secrets of both parties and without yielding these secrets) much more inspirating. However , the author does not know of any reduction between the biased coin used here and the symmetric biased coin suggested in [LMR]; there are too many differences in the setting, conception and implementation!

[1] As in Sec.3 it will happen that other users know part of the sequence but only one user (its holder) knows all of it, provided he follows the cip which reveals parts of it properly.

4.1. Secure Experimental Sequences

Although we have given some clues to explain how we intend to construct an experimental sequence we would like to define it in a more general manner so that one can analyze the cip we are about to present in the next subsection independently of our implementation of the secure experimental sequence.

Definition: A *Secure Experimental Sequence* (SES) of user X (hereafter referred to as its *holder*) testable by user Y (hereafter referred to as its *examiner* is a pair of a sequence and an anti-replay part[2] such that:

(D1) The sequence gives information about the identity of X:

Given the entire sequence the entropy of the message which consists of X's name is very low.

The change , in the entropy of the message which yields X's name , caused by revealing one entry is very small (independent of what the value of this entry is).

Remark: Condition (D1) can be understood more easily in terms probability : "Given the sequence one can guess with very high probability of success the identity of X. This probability does not change much if one is given one entry."

Unfortunately we found it hard to prove that our implementation satisfies (D1) when stated in terms of probability. For details on entropy,a-posteriori entropy and average a-posteriori entropy see any information theory textbook (e.g. Abramson[A]).

(D2) When given an entry in a sequence Y can tell whether it belongs to a sequence which satisfies (D1).

(D3) X does not get information on the identity of Y by knowing that a sequence testable by Y is being tested.

(D4) Y can tell whether the sequence and the anti-replay part belong to the same SES.

Remark: Note that Y can tell whether the anti-replay part is transmitted to him by X. Thus if (D4) hold Y can tell whether the sequense is transmitted to him by X.

[2] as in Sec. 3

4.2. Sketch of the Concurrent Identification Protocol

(The parties to the cip will be denoted A and B)
(0) A notifies B which of B's SESs he would like to examine.
B acts symmetrically w.r.t A's SESs.
(1) A checks whether he is communicating with the legitimate holder of the SES (i.e. B).
B acts symmetrically.
(This is done by testing the anti-replay part of the SES similarly to the way it was done in the cip of Sec. 3.)
(2) for i=1 to q (the number of entries in a SES) do begin
A transmits the i-th entry of his SES to B.
B acts symmetrically.
end

4.3. Analysis of the Protocol

Under the assumption that there exist SESs in the network it is straightforward to prove that the cip presented above is secure,namely:

Theorem 4: If a user ,U plays the above cip properly then the following hold:

(1) In any phase during the execution of the protocol,if for U's counterpart the entropy of U's name is e then for U the entropy of what is claimed to be his counterparts name is very close to e.
(2) If U's counterparts is honest U will find out his identity.
(3) If U's counterpart is impersonating then with high probability $(1 - 2^{-n})$ U will find this out before reaching a stage in which he has revealed any information about his identity .

proof: (1) follows from step(2) of the cip combined with (D1),(D2) and (D3) of the SES definition . (2) follows from step(2) combined with (D1). (3) follows from step(1) and (D4).

Q.E.D

4.4. Implementing a SES

We will develop a SES step by step: showing simple versions and the reasons they fail and only at the end prove that the final version meets the conditions of the definition of a SES. As hinted at the beginning of Sec. 4 the experimental sequence will consist of blocks-of-entries associated with the bits of its holder's name.

4.4.1. A Simple ES and its Failure

Let $(x_1, x_2, ..., x_l)$ denote the bit representation of X's name. Denote by $\overline{x}$ the complement of x (i.e. $1 - x$). Let δ be a rational number such that $0.5 < \delta < 1$ and m be an integer.

Definition: A (δ,m)-ES of X [held by X] consists of l binary blocks each of length m, such that in the i-th block there are δm bits equal to x_i (and $(m - \delta m)$ bits equal to $\overline{x}_i$).

Note that this ES has some desirable properties: If a non-holder who does not know X's name is given bits of this ES then the probability that he can guess X's identity will gradually increase.(Note that reading only a few bits in a block does not help much if δ is slightly bigger than 0.5 but reading enough bits in a block ,provided it is long enough will disclose the corresponding bit in X's name.)

Unfortunately, it is not guaranteed that X will reveal the bits of his ES. He can cheat without being caught and even without arousing his counterpart's suspicions. To prevent the holder from cheating, trusted signatures to the bits of the ES must be provided. The bits should remain unknown also to the person who signed them but he must have a way of verifying that he is signing a (δ,m)-ES; thus the blind signatures proposed by Chaum[C] are of no help.

4.4.2. A Signed ES and its Failure

An idea which we found useful is to let the person who signs the bits of the ES create them ,obliviously, by executing instances of a δ-OT protocol with the ES's holder (or to be exact: future holder). The δ-OT is a generalization of the OT described in Appendix A . However, in the δ-OT the two messages are not equally likely to be received; one message will be received with probability δ while the other with probability $(1 - \delta)$. An explicit definition of the δ-OT as well as its implementation can be found in Appendix B.

Definition: A Y-Signed (δ,m)-ES of X consists of l blocks each consisting of m elements such that for a non-holder who knows X's name the a-priori probability that the j-th element in the i-th block is $D_Y(i,j,x_i)$ is equal to δ. Otherwise this element is $D_Y(i,j,\overline{x}_i)$.

A pre-protocol for the generation of a Y-signed (δ,m)-ES follows:

`for i=1 to` l `do;for j=1 to` m `do`
Y `sends the pair` $(D_Y(i,j,x_i), D_Y(i,j,\overline{x}_i))$ `via` δ`-OT to` X.
`(with probability` δ X `gets the l.h.s element.)`

Before being tempted to present a cip which uses the product of this pre-protocol let us consider the following difficulties:

(Problem 1) If during the cip Y asks X to examine X's Y-signed ES X will immediatly know that his counterpart is Y. (Unless all the users are willing to trust some user to generate good (δ,m)-ESs signed by this user. This idea can be extended to a trusted subset. It should be noted that these users are only trusted not to collaborate with any holder, different from the trusted center described in Sec. 3 which was also trusted not to reveal information to the examiners. However, we will not deal with this case here.)

4.4.3. An Anonymously Signed ES and its Failure

The solution to the above (Prob.1) is simple. User Y introduces a virtual user, Y', and executes the pre-protocol using the identity of the virtual user to produce Y'-signed ESs to all the users (including himself for security reasons). Y is called the creator of Y'. **The fact that Y is the creator of Y' must be kept secret.** Thus, Y can not execute the cip twice using the same virtual user. This difficulty is overcome by letting Y create many virtual users, distribute ESs signed by each of these virtual users to all the real users in the net and when executing the cip impersonate as a different virtual user in every instance.

To sum up: instances of the pre-protocol will be played between each of the virtual users, created by each real user, and each of the real users. These instances will provided each real user with ESs signed by virtual users the identities of their creators are unknown to him.

We are now left with the last problem:

(Problem 2) Users may collaborate to forge their Y'-signed ESs. This can be done by revealing to each other what are the entries in their Y'-signed ESs. (If enough users collaborate each will know both possible values for each entry. This will allow each of them to construct a Y'-signed ES which does not have any bias , or an Y'-signed ES which leads to anybody else.)

4.4.4. The Numbered Anonymous Signed ES

To solve this difficulty we must provide a method by which Y' can stamp all the ESs he distributes such that each bears a different stamp **but without knowing which stamp he has put on the ES he has generated for a given user.** If the stamp is randomly chosen from a domain of d stamps and d is large enough ($d = |N|^{2+\epsilon}$, where $\epsilon > 0$ will do) then with very high probability no two users in N hold a Y'-signed ES bearing the same stamp. The pre-protocol will assure Y that the Y'-signed ES he generates for X bears a number chosen at random from the set $\{1, 2, .., d\}$ **but** will not allow Y to guess this number successfully (with probability higher than d^{-1}). The details of the complete implementation of the pre-protocol , by which numbered Y'-signed (δ,m)-ES are distributed, are given in Appendix C.

4.4.5. Analysis of The Numbered Anonymously Signed ES

We are now ready to show that there is no bug in our proposal, namely:

Theorem 5: The Numbered Anonymously Signed ES (hereafter referred to as NAS-ES) is a SES.

proof: We will show that the problem of determining x_i given t elements in the i-th block of X's NAS-ES corresponds to the problem of determining the key given t outputs in a Binary Symmetric Channel (BSC).

The inputs of the BSC as well as its outputs and keys are elements of the set $\{0, 1\}$. The inputs are independently chosen according to some a-priori fixed probability. (The input is 1 with probability p and 0 with probability $1 - p$) The key is chosen randomly with equal probability (to be either 0 or 1) and remains fixed throughout

the communication process. The j-th output is equal to the j-th input if the key in use is 1 and is equal to the complement of the j-th bit if the key in use is 0 (see Figure 1). The problem is to determine which key has been used given t outputs of the communication which used it. For more details see Abramson[A].

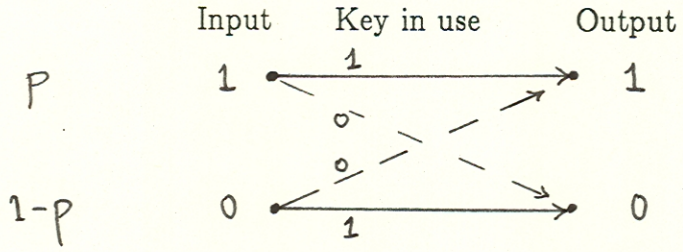

Figure 1: A Binary Symmetric Channel

Note that the j-th output of the BSC is equal to the key in use iff the j-th input is 1. We now reduce the problem of determining x_i given t elements in the i-th block of X's NAS-ES to the problem of determining the key given t outputs in a BSC when the a-priori probability 1 is transmitted is δ and the a-priori probability 0 is transmitted is $1 - \delta$:

> The value of x_i corresponds to the key; the elements of the i-th block which have been examined correspond to the outputs; while the bits which denote whether these examinations were "good"(i.e. gave the value of x_i) correspond to the inputs.

Thus the analysis of the entropy of x_i (given t entries in the i-th block) is identical to the analysis of the entropy of the key in the BSC (given t outputs). This analysis was carried out by Shannon[S] (who also introduced the problem). In rough words it suits our intuition that the amount of information, about the NAS-ES holder, revealed by a single experiment is very small if δ is close to 0.5 . (e.g. $\delta = 0.55$ gives an entropy advantage of no more than 0.01 to a party which stops the execution of the protocol.) Also, after enough examinations one can practically know the identity of the NAS-ES holder. (A choice of $m = O((\delta - 0.5)^{-1})$ will do.)

Note that the only information the examiner gets by examining one entry of the i-th block of the NAS-ES is the value of one experiment on x_i. Thus, (D1) follows.

(D2) follows from the fact that the NAS-ES was signed by the examiner (when impersonating) some virtual user he created and from the fact that (with high probability) there is no other NAS-ES which bears the same serial number. (D3) follows from the fact that the NAS-ES yields only the identity of the virtual user who signed it, **but not the identity of the user** who created this virtual user. (D4) follows from the fact that the anti-replay part and the sequence bear the same serial number and that there exist no other NAS-ES which bear this number.

Q.E.D.

4.5. The Concurrently Identification Protocol

(The parties to the cip will be denoted A and B)
(0) A choses one of his unused virtual users ,denoted A'.
A transmits A' and $D_{A'}(A')$ to B.
B acts symmetrically transmitting B' and $D_{B'}(B')$.
(Denote by ES_A the B'-signed SES held by A
and by ES_B the A'-signed SES held by B.)
(1) for i=1 to n do begin
A sends to B' (B) one element of AR_i of ES_A.
B sends to A' (A) one element of AR_i of ES_B.
end
(2) for $j = 1$ to m
for $i = 1$ to l do begin
A transmits the j-th element of the i-th block of ES_A to B.
B transmits the j-th element of the i-th block of ES_B to A.
end

5. Acknowledgements

I would like to thank Prof. Shimon Even for proposing me the problem, for many helpful discussions and especially for help in the analysis of the cip presented in Sec. 4.

I would also like to thank Prof. Adi Shamir for the discussion which led to the cip presented in Sec. 3.

I am very grateful to Tom Tedrick for carefully reading the manuscript, finding many errors and giving me feedback.

6. References

[A] Abramson,N., *Information Theory and Coding*, M_cGraw-Hill,1963, pp. 100-105.

[C] Chaum,D., "Blind Signatures for Untraceable Payments", in *Advances in Cryptology:Proceedings of Crypto82*, (Chaum,D. et al. editors), Plenum Press, 1983, pp. 199-203.

[DH] Diffie,W., and Hellman,M.E., "New Directions in Cryptography", *IEEE Trans. on Inform. Theory*,Vol. IT-22,No. 6,November 1976, pp. 644-654

[EGL] Even,S., Goldreich,O., and Lempel,A., "A Randomized Protocol for Signing Contracts", in *Advances in Cryptology:Proceedings of Crypto82*, (Chaum,D. et al. editors), Plenum Press, 1983, pp. 205-210

[*EGL'*] Even,S., Goldreich,O., and Lempel,A., "A Randomized Protocol for Signing Contracts", TR No. 233, Computer Science Dept., Technion, Haifa, Israel, February 1982

[LMR] Lubi,M., Micali,S., and Rackoff,C., "How to Simultaneously Exchange a Secret Bit by Flipping a Symmetrically-Biased Coin", *proceedings of the 24th IEEE Symp. on Foundation Of Computer Science*, 1983, pp. 11-21

[NBS] National Bureau of Standards, Data Encryption Standard, *Federal Information Processing Standards*, Publ. 46, 1977

[R] Rabin,M.O., "How to Exchange Secrets by Oblivious Transfer", Technical memo TR-81, Harvard Center for Research in Computing, (1981).

[RSA] Rivest,R.L., Shamir,A., and Adleman,L., "A Method for Obtaining Digital Signature and Public Key Cryptosystems", *Comm. of the ACM* ,Vol.21, February 1978, pp. 120-126

[S] Shannon,C.E., "Communication Theory of Secrecy Systems",*Bell Syst. Jour.* 28, October 1949, pp. 656-715

[T] Tedrick,T., "How to Exchange Half a Bit", to appear in the proceedings of *Crypto83*

7. Appendices

7.1. Appendix A : An Implementation of OT

Assume S wants to transfer to R exactly one of the messages M_1 and M_2,such that:

(1) R can recognize both M_1 and M_2
(e.g. they are signatures to known messages).
(2) If S is honest then R gets M_1 with probability one half.
For S the a-posteriori probability that R got M_1 remains one half.
(3) If S tries to cheat, R will detect it with probability at least one half.

An implementation of this transfer proceeds as follows:

(0) S chooses ,randomly, two pairs (E_1, D_1) and (E_2, D_2) of encryption-decryption algorithms of the PKCS.
R chooses ,randomly, a key K for the conventional cryptosystem F.
(1) S transmits E_1 and E_2 to R.
(2) R chooses ,randomly, $r \in \{1,2\}$ and transmits $E_r(K)$ to S.
(3) S computes $K'_i = D_i(E_r(K))$,for $i \in \{1,2\}$.
S chooses ,randomly, $s \in \{1,2\}$ and transmits $(F_{K'_1}(M'_1), F_{K'_2}(M'_2), s)$ to R, where $M'_s = M_1$ and $M'_{3-s} = M_2$.

Remarks:

(1) Assuming that K looks like random noise and that E_1,E_2 have the same range, S can not guess with probability of success greater than one half which of the K'_i's,computed by him is the K choosen by R.

(2) By the assumption that the PKCS is secure and that its instances are free of any relation other than the cancellation of encryption by the corresponding decryption : R can not compute $K'_i \neq K$.

(3) By the assumption that F is secure R must have K'_i in order to read M'_i.

(4) By (1) and (2) if S is not cheating then R can read M'_i iff $i = r$. Thus, he can detect cheating by S with probability one half.

(5) In the RSA[RSA] scheme, distinct E_i's may have different ranges. However, this difficulty can be overcome (see [EGL']).

(6) One can use a one-time pad instead of the conventional cryptosystem F.

7.2. Appendix B : An Implementation of δ-OT

Let δ be a rational number such that $\delta=(p/q)$, where p and q are integers and $q<p<2q$. A δ-OT allows user S to transfer exactly one of the recognizable messages M_1 and M_2 to R such that: (1) R gets M_1 with probability δ provided S plays the protocol properly. (2) for S the a-posteriori probability that R got M_1 remains δ. (3) if S tries to prevent R from geting M_1 and R gets it with probability r then R can detect that S is cheating with probability at least (δ-r).

An implementation of the δ-OT proceeds as follows:

(0) S chooses ,randomly, a set of q pairs $\{(E_i, D_i)\}_{i=1}^{q}$
of encryption-decryption algorithms for the PKCS.
R chooses ,randomly, a key K
for the conventional cryptosystem F.
(1) S transmits $E_1, E_2, \ldots, E_q$ to R.
(2) R chooses ,randomly, $r \in \{1, 2, .., q\}$ and transmits $E_r(K)$ to S.
(3) S computes $K'_i = D_i(E_r(K))$,for $i \in \{1, 2, .., q\}$.
S chooses ,randomly, a subset $S^{(p)} \subseteq \{1, 2, .., q\}$ of cardinality p
and transmits $(F_{K'_1}(M'_1), F_{K'_2}(M'_2), \ldots, F_{K'_q}(M'_q), S^{(p)})$ to R,
where $M'_i = M_1$ if $i \in S^{(p)}$
and $M'_i = M_2$ otherwise.

Remark: Note that this protocol is a generalization of the protocol presented in Appendix A (which implement a (1/2)-OT). All the remarks which follow the protocol presented in Appendix A can be generalized to apply for the protocol presented here.

7.3. Appendix C : An Implementation of the Pre-Protocol

The following protocol allows the virtual user Y' to provide user X with a Y'-signed ((p/q),m)-ES which bears a serial number between 1 and k.

The protocol consists of an OT which transfer one message out of a set of k recognizable messages. These messages are determined by instances of a δ-OT subprotocol. We remind the reader that l denotes the number of bits in a user's name while m denotes the number of experiments conducted on such a bit.

(0) Y' chooses, randomly, a set of k encryption-decryption pairs $\{(E_i, D_i)\}_{i=1}^{k}$ for a PKCS.
X chooses, randomly, a key K for the conventional cryptosystem F.

(1) Y' transmits $E_1, E_2, ..., E_k$ to X.

(2) X chooses, randomly, $r_0 \in \{1, 2, .., k\}$ and transmits $E_{r_0}(K)$ to Y'.

(3) Y' computes $K'_i = D_i(E_{r_0}(K))$, for $i \in \{1, 2, .., k\}$.
Y' chooses a permutation ρ on $\{1, 2, .., k\}$.
[The following steps determine the messages $M'_1, M'_2, .., M'_k$ one of which will be read by X .]

(4) for g=1 to l do
for h=1 to m do begin
[Determine the value of the h-th entry in the g-th block of the Y'-signed $((p/q), m)$-ES.]

(4.0) Y' chooses, randomly, a set $\{(E_i^{g,h}, D_i^{g,h})\}_{i=1}^{q}$, of q encryption-decryption pairs for a PKCS.
X chooses, randomly, a key $K^{g,h}$ for F.

(4.1) Y' transmits $E_1^{g,h}, E_2^{g,h}, .., E_q^{g,h}$ to X.

(4.2) X chooses, randomly, $r_{g,h} \in \{1, 2, .., q\}$ and transmits $E_{r_{g,h}}^{g,h}(K^{g,h})$ to Y' .

(4.3) Y' computes
$$\overline{K}_i^{g,h} = D_i^{g,h}(E_{r_{g,h}}^{g,h}(K^{g,h})), \text{ for } i \in \{1, 2, .., q\}.$$
Y' chooses, randomly, a subset
$$S_{g,h} \subseteq \{1, 2, .., q\} \text{ of cardinality } p.$$
Y' computes
$$N_{g,h}^{j,1} = D_{Y'}(\rho(j), g, h, x_g), N_{g,h}^{j,0} = D_{Y'}(\rho(j), g, h, \overline{x}_g),$$
for $j \in \{1, 2, ..., k\}$.

end

[DEFINITIONS] for $j \in \{1, 2, ..., k\}$
for $i \in \{1, 2, .., q\}, g \in \{1, 2, .., l\}$ and $h \in \{1, 2, .., m\}$ define
$M_{g,h,i}^{(j)} = N_{g,h}^{j,1}$ if $i \in S_{g,h}$ and $M_{g,h,i}^{(j)} = N_{g,h}^{j,0}$ otherwise;
$\overline{M}_{g,h}^{(j)} = (F_{\overline{K}_1^{g,h}}(M_{g,h,1}^{(j)}), F_{\overline{K}_2^{g,h}}(M_{g,h,2}^{(j)}), .., F_{\overline{K}_q^{g,h}}(M_{g,h,q}^{(j)}), S_{g,h})$.
for $i \in \{1, 2, .., n\}$ define
$AR_i^{(j)} = (D_{Y'}(\rho(j), S'_i), D_{Y'}(\rho(j), S''_i))$
for $i \in \{1, 2, .., l\}$ define
$B_i^{(j)} = (\overline{M}_{i,1}^{(j)}, \overline{M}_{i,2}^{(j)}, .., \overline{M}_{i,m}^{(j)})$
Define $M'_j = (AR_1^{(j)}, AR_2^{(j)}, .., AR_n^{(j)}; B_1^{(j)}, B_2^{(j)}, .., B_l^{(j)})$.
[end of definitions]

(5) Y' transmits $(F_{K'_1}(M'_1), F_{K'_2}(M'_2), .., F_{K'_k}(M'_k), \rho)$ to X.

REMARKS

(1) The S'_is and the S''_is are arbitrary messages which are independent of X.

(2) Note that X can read exactly one of the M'_is (can only read M'_{r_0} which bears the serial number $\rho(r_0)$). Thus X can read $\{AR_i^{(r_0)}\}_{i=1}^{n}$ and $\{\overline{M}_{g,h}^{(r_0)}\}_{g=1,h=1}^{l,m}$. For every $g \in \{1,2,..,l\}$ and $h \in \{1,2,..,m\}$ X can read only one of the $M_{g,h,i}^{r_0}$s (i.e. $M_{g,h,r_{g,h}}^{r_0}$). Note that the messages read by X constitue a Y'-signed $((p/q),m)$-ES which bears a random serial number between 1 and k.

(3) The volume of communication in this subprotocol is of order $k(lmq+n)$ while the number of transmissions is of order lm and can be made constant by interleaving the iterations of step(4).