On Key Distribution Systems

Y. Yacobi
Bellcore, 445 South St.
Morristown NJ 07960
yacov@bellcore.com

Z. Shmuely
Computer Science Department
Technion, Haifa 32000, Israel

Zero Knowledge (ZK) theory formed the basis for practical identification and signature cryptosysems (invented by Fiat and Shamir). It also was used to construct a key distribution scheme (invented by Bauspiess and Knobloch); however, it seems that the ZK concept is less appropriate for key distribution systems (KDS), where the main cost is the number of communications. We propose relaxed criteria for the security of KDS, which we assert are sufficient, and present a system which meets most of the criteria. Our system is not ZK (it leaks few bits), but in return it is very simple. It is a Diffie-Hellman variation. Its security is equivalent to RSA, but it runs faster.

Our definition for the security of KDS is based on a new definition of security for one-way functions recently proposed by Goldreich and Levin. For a given system and given cracking-algorithm, I, the cracking rate is roughly the average of the inverse of the running-time over all instances (if on some instance it fails, that inverse is zero). If there exists a function $s: N \rightarrow N$, s.t. for all I, the cracking-rate for security parameter n is $O(1)/s(n)$, then we say that the system has at least security s. We use this concept to define the security of KDS for malicious adversary (the passive adversary is a special case). Our definition of a malicious adversary is relatively restricted, but we assert it is general enough for KDS. This restriction enables the proof of security results for simple and practical systems. We further modify the definition to allow past keys and their protocol messages in the input data to a cracking algorithm. The resulting security function is called the "amortized security" of the system. This is justified by current usage of KDS, where the keys are often used with cryptosystems of moderate strength. We demonstrate the above properties on some Diffie-Hellman KDS variants which also authenticate the parties. In particular, we give evidence that one of the variants has super-polynomial security against any malicious adversary, assuming RSA modulus is hard to factor. We also give evidence that its amortized security is super-polynomial. (The original DH scheme does not authenticate, and the version with public directory has a fixed key, i.e. zero amortized security.)

1. Introduction

Zero Knowledge theory [GMR] formed the basis for some practical identification and signature cryptosystems, most notably the Fiat-Shamir [FS] identification scheme. Recently, a zero-knowledge key distribution system was proposed [BK]. The advantage of a zero knowledge system is that no information leaks from the system; therefore, repeated use of the system does not make it less secure. This is true for the case of a malicious adversary, too.

However, it seems that the ZK concept is less appropriate for key distribution systems (KDS), where the main cost is the number of communications. We argue that by

G. Brassard (Ed.): Advances in Cryptology - CRYPTO '89, LNCS 435, pp. 344-355, 1990.

allowing some insignificant leak of information we can achieve simpler systems.

We propose relaxed criteria for the security of KDS, which we assert are sufficient, and present a system which meets most of the criteria. Our system is not ZK (it leaks few bits), but in return it is very simple. It is a Diffie-Hellman variation. Its security is equivalent to RSA, but it runs faster.

Our definition for the security of Key-Distribution Systems is based on a new definition of security for one-way functions recently proposed by Goldreich and Levin [GL]. For a given system and given cracking-algorithm, I, the cracking rate is roughly the average of the inverse of the running-time over all instances (if on some instance it fails, that inverse is zero). If there exists a function $s:N\rightarrow N$, s.t. for all I, the cracking- rate for security parameter n is $O(1)/s(n)$, then we say that the system has at least security s. We use this concept to define the security of KDS for a malicious adversary (the passive adversary is a special case). Our definition of a malicious adversary is relatively restricted (compared to [GMW]), but we assert it is general enough for KDS. This restriction enables the proof of security results for simple and practical systems. We further modify the definition to allow past keys and their protocol messages in the input data to a cracking algorithm (similarly to known-plaintext attack on cryptosystems). The resulting security function is named the "amortized security" of the system. This is justified by current usage of KDS, where the keys are often used with cryptosystems of moderate strength, e.g. DES. Ideally we would like the amortized-security to equal the security, and be super-polynomial.

There is a trivial solution to the problem of achieving key-distribution together with party authentication using public-key cryptosystems. This solution has a disadvantage though, when using RSA or its derivatives. We need a distinct modulus for each user, and this complicates computations. A potential family of KDS which can use a common modulus is the Diffie-Hellman [DH] scheme and its variants. The original Diffie-Hellman scheme does not authenticate, and the version with public directory (see section 3) has a fixed key, i.e. zero amortized security. We propose a simple and practical KDS (two exponentiations and one transmission per party) which authenticates the parties, and we give evidence for its super-polynomial security for any malicious adversary, assuming RSA modulus is hard to factor. We also give evidence that its amortized security is super polynomial.

In section 2 we define our security criteria, in section 3 we present some variants of the Diffie-Hellman KDS, and show their pitfalls, and in section 4 we show a relatively secure Diffie-Hellman variant. In the appendix, we modify the Goldreich-Levin definition of security for cryptosystems, taking into account the long neglected fact that messages of low probability may often be the most important ones (according to information theory they have high information content).

While some encryption schemes offer very good protection to every bit of information [GM], it may happen that the same protection value could be achieved with simpler systems, which concentrate on protecting the most important messages. This is captured by our new definition.

Some other KDS were proposed in [G], [KO], [MTI]and [O], with new features, but with no proofs of security. Bauspiess and Knobloch [BK] published a zero-knowledge KDS,

but their system is more complicated than ours. We argue that at the cost of leaking few bits of our secrets we buy simplicity.

2. Proposed Criteria

2.1 General

We first give the Goldreich-Levin [GL] definition of security for one-way functions. We use their notation. Let S be the set of finite and Ω of infinite strings over {0,1}, and let $S_n \subseteq S$ be the set of strings of length n. Let $N = \{0,1,..\}$. $E_x f(x)$ denotes the expected value of f (for a given distribution function $x=d(r)$). Let $I(\omega,y)$ be a probabilistic algorithm which attempts to invert a function f, i.e. to recover $x \in S$ from $y=f(x)$, using $\omega \in \Omega$. Let $T_I(\omega,y)$ be I's running time. Let I's "success bit" $S_{I,f}(\omega,x)=1$ if $I(\omega,f(x))=x$ and 0 otherwise. The inverting rate $R_{I,f,d}(n)=E_{r,\omega}(S_{I,f}(\omega,d(r))/T_I(\omega,f(d(r))))$.

Definition 1 [GL]: A function f is called *one-way* on distribution d with security $s:N \rightarrow N$ if $R_{I,f,d}(n)=O(1)/s(n)^{\varepsilon}$, for some $\varepsilon>0$ and all probabilistic algorithms I.

We next define our general KDS, restricting our attention to two-party systems. These (honest) parties try to establish a session-key, to be used later in some crypto-system. Each of the parties has a secret key and a public key. The parties exchange messages according to some protocol. At the end of the protocol they compute the session-key. That is

A 2 *party Key Distribution System* (*KDS*) is defined by the following i/o relation:

Input: *clear*: $P=(P_1,P_2), x=(x_1,..x_q)$; *secret*: $U=(U_1,U_2)$,

Output: *secret*: $K=f_1(P,U_1,x)=f_2(P,U_2,x)$

For security parameter n, each of the variables P_i,U_i,x_i,k is in S_n. P is the set of public keys, U is the set of secret keys, and x is the ordered set of messages exchanged between the parties during the execution of the protocol. Usually q is very small. K is the resulting session-key. f_1 and f_2 are polynomial time functions, mapping binary strings to binary strings.

The distribution of P,U,x is determined by some multidimensional distribution function d with random variable r, $r \in N$, as input.

2.2 Passive Adversary

We give here a variant of Definition 1 which we tailor for KDS (see above). Let $I(\omega,P,x)$ be any probabilistic algorithm trying to compute K. I models an adversary that tries to crack the system. Let $T_I(\omega,P,x)$ be I's running time. Let I's "success bit" $S_{I,f}(\omega,P,S,x)=1$ if $I(\omega,P,x)=K$ (and 0 otherwise). The *cracking rate* for security parameter n is $R_{I,f,d}(n)=E_{r,\omega}(S_{I,f}(\omega,P,U,x)/T_I(\omega,P,x))$, where P_i,U_i,x_i,k are in S_n.

Definition 2: A KDS has at least *security* $s:N \rightarrow N$ against passive adversaries if $R_{I,f,d}(n)=O(1)/s(n)$ for all probabilistic algorithms I.

Note that we defined a lower bound on the security, not "exact" security. Also, we omited the ε, since we want to distinguish between security functions which are

polynomially related (however, this point is not significant).

2.3 Malicious Adversary

Goldreich, Micali and Wigderson [GMW] treat the problem of finding a secure protocol for carrying out any feasible distributed protocol. They define a malicious adversary as a machine that can deviate from its prescribed program in any possible action, and describe a way to transform any protocol into a protocol which is secure against any minority of malicious adversaries.

In KDS protocols we have three players, namely, the two (honest) parties, trying to authenticate each other, and establish a session key, and the adversary, playing in the middle, trying to compute the key, in the the case of a passive adversary, or trying to establish some key with each of the parties, pretending to be his counterpart, in the case of a malicious adversary. So, since the adversary is a minority, by [GMW] we know that a secure polynomial time protocol exists (i.e. such that it overcomes any possible malicious adversary).

Our aim, therefore, is to use the fact that ours is a special case to achieve a very efficient protocol which overcomes any malicious adversary. In section 4 we show a KDS protocol which requires just two exponentiations for each of the parties, and the proof of its security is very simple.

A malicious adversary can interfere in a KDS protocol in various ways, he can initiate a protocol, cut the line of a user and connect himself instead, waiting to receive some initiative, he may initiate a KDS protocol with two sides simultaneously, or interfere between two honest parties trying to run KDS protocol.

Let $(\underline{x}_1,..\underline{x}_q)$ be the ordered set of messages exchanged between a malicious adversary z and an honest party in the course of KDS protocol, and let $\underline{x}^i=(\underline{x}_1,..\underline{x}_i)$, for $i=1,...,q$. If the honest party initiates then $\underline{x}_1=x_1$. If z initiates then $\underline{x}_1=h_1(P,U_z,x_1)$, where x_1 is a legitimate protocol message that the honest party could get when communicating with the party he assumes he communicates with, and h_1 (and later we use h_i) is any probabilistic polynomial time algorithm. h may have any other non-secret input. To simplify denotations we omitted it.

Definition 3: A *malicious adversary* z interferes with KDS protocols in such a way that a legitimate party ends up with protocol messages $\underline{x}=\underline{x}^q$, where for $i=1,2,...,q$, $\underline{x}_i=h_i(P,U_z,\underline{x}^{i-1})$. Accordingly, the legitimate party computes $\underline{K}=f(P,U,\underline{x})$, instead of the key K. The attack is *successful* if z can efficiently compute $\underline{K}$.

The case in which z interferes between two legitimate parties trying to carry out a KDS protocol is called "two-way impersonation attack." See [Y] for an example of such a successful attack. Note that, in this case, the malicious adversary may compute two distinct keys, one with each of the honest parties. He doesn't even have to use the same functions $h_i(.)$ in both directions.

The definition of *security* for malicious adversary is the same as for passive adversary with one modification, namely, the function I is replaced by $\underline{I}(\omega,P,\underline{x})$ trying to compute $\underline{K}$.

Clearly, the passive adversary is a special case of the malicious adversary for which $\underline{x}=x$ and $\underline{K}=K$.

2.4 Amortized Security

The *amortized security* of a KDS is roughly the complexity of its cracking problem given a history of keys and their respective protocol messages x. It is very similar to the definition of a "known cleartext attack" for cryptosystems. The motivation for this definition in the context of KDS stems from current usage of public KDS, where the resulting key is used in conventional cryptosystems, like DES, which are of moderate strength. Ideally we would like the complexity of this problem to be independent of the extra information contained in the old keys and their sessions. We consider amortized security for passive and malicious adversaries. The definitions of security are the same as before, only the algorithm I, which tries to crack the system, is modified to get more information, old keys and their sessions.

3. Some Diffie-Hellman variations

Next, we discuss some Diffie-Hellman (DH) KDS variations, which authenticate the parties, and show that their amortized security is low. In all the DH variations presented in this paper the parties should compute identical keys, if nobody cheated. Authentication is completed by trying to use the resulting key on recognizable messages (e.g., a message appended with 20 zeros).

3.1 The original Diffie-Hellman system

The original DH KDS [DH] has a variation which enables authentication of the parties. In this system there is a public trusted read-only directory in which the name, phone number, and public key of each participant appears. The public-key of participant i is $P_i \equiv \alpha^{x_i} \ mod \ m$, where x_i is randomly chosen by i, and known only to i. In the original scheme m was a prime, and α a generator in $GF(m)$. Let Zm denote the ring of integers modulo m, for any m. Recently some other groups were suggested for this application, where m is a composite, and α generates a large enough fraction of Zm (e.g., a quarter of it). See for example [S] and [M].

When j wants to communicate secretly with i, he computes $K_{j,i} \equiv P_i^{x_j} \ mod \ m$, and tells i in the clear that he wants to secretly communicate with her. Party i computes $K_{i,j}$ likewise, so that $K_{i,j}=K_{j,i}$. Clearly, this system can authenticate and establish a session key, but, since whenever two specific parties i and j establish a key, they end up with the same key, this system has zero amortized security for a passive adversary.

3.2 Time dependent Diffie-Hellman variation

Here we assume that at each moment there is time t known to every participant. When i wants to establish a session key with j she computes $K_{i,j,t} \equiv (K_{i,j})^t \ mod \ m$, where $K_{i,j}$ is as before, tells j in the clear that she, i, wants to communicate with him, and j computes $(K_{j,i})^t$. As before, $K_{i,j,t}=K_{j,i,t}$, if nobody cheated.

Given two keys $K_{i,j,t}$ and $K_{i,j,t+1}$ one can easily compute any key $K_{i,j,t'} \equiv (K_{i,j,t+1} \cdot K_{i,j,t}^{-1})^{t'} \bmod m$.

If the adversary has $K_{i,j,t+\delta}$, for some small δ, instead of $K_{i,j,t+1}$, then he still can compute every key of the form $K_{i,j,\delta \cdot p}$, for every integer p. Therefore, this system has a negligible amortized security for a passive adversary.

3.3 Randomized Diffie-Hellman variation

Let $K_{i,j}$ be as before. Here the parties randomly choose R_i and R_j, and exchange these values in the clear. They now compute $K'_{i,j} \equiv (K_{i,j})^{R_i+R_j} \bmod m$.

A malicious adversary, who knows one key $K'_{i,j}$ and wants to impersonate j as a receiver, can disconnect j, and connect himself instead. Whenever i initiates a call to j, sending her R'_i, the adversary responds with $R'_j = R_i + R_j - R'_i$ (this is not a modular operation, since the adversary doesn't know $\phi(m)$). The result is a "new" key which equals the known one. Therefore, this system has zero amortized complexity for a malicious adversary.

4. A relatively secure Diffie-Hellman variation

In this section we describe another Diffie-Hellman variation which authenticates the parties, and we give evidence that its security is super-polynomial for passive and malicious adversary. We do not know the status of its amortized security, but believe it to be super-polynomial.

4.1 Description

Shmuely [S], and later McCurley [M], gave evidence that the Composite Diffie-Hellman (CDH) scheme (i.e. DH scheme with RSA-like modulus) is hard to break, in the sense that if there was an efficient algorithm which breaks a fixed fraction δ, $0<\delta<1$, of the instances, then we could factor the modulus with high probability in time proportional to δ^{-1}. If the system uses an "RSA modulus," believed to be hard to factor "almost everywhere," we conclude that it is probably impossible to break a fixed fraction δ, for any δ, of the instances of CDH. The proof may be extended to any $n^{-O(1)}$ fraction, where n is the problem size in bits. In this case we claim that if there exists an efficient algorithm which cracks $n^{-O(1)}$ of the instances of CDH, then it could be used to efficiently factor the modulus with high probability.

With the proper (now conventional) assumptions about the difficulty of factoring the modulus (RSA modulus), one can show that CDH KDS has a super-polynomial security. We base our system on this CDH system, and inherit this important property for passive and malicious adversaries.

The system

Each user i possesses a *public key* P_i, and a *secret key* S_i, $P_i, S_i \in [0, m)$, where $P_i \equiv \alpha^{S_i} \bmod m$, m is an "RSA modulus" and α is a base element which generates a large enough fraction of Zm, the ring of integers modulo m. Suppose that two legitimate users of this system i and j want to establish session key $K_{i,j}$. They follow this protocol:

begin

i selects a random number $R_i \in [0,m)$ and sends the message $X_i=R_i+S_i \in [0,2m-1)$ to j, who reciprocates likewise computing and sending X_j to i,

i computes $K_{i,j} \equiv (\alpha^{X_j} \cdot P_j^{-1})^{R_i} \equiv \alpha^{R_i R_j} \ mod\ m$ and j reciprocates likewise computing $K_{j,i}$, which equals $K_{i,j}$ if nobody cheated.

end

Note that in this system none of the users needs to know the factorization of the modulus m. However, the central authority, which publishes the public directory, and is responsible for its integrity, must be able to prove that indeed m is a legitimate RSA modulus. Galil, Haber and Yung [GHY] showed a direct, efficient method to prove in zero-knowledge that a given number m is of the form pq, where p and q are primes. In principle all the other properties of our modulus, i.e. that $p-1=2p'$, and that $p-q>b$, for some given b, can be proven in zero-knowledge, since the corresponding decision problems are in NP. However, we know of no direct, efficient proof of these properties. To implement the proofs of [GHY], the central authority must know the factorization of m.

4.2 Distributional problems

A distributional problem is a decision problem with probability of appearance attached to each of the instances. For a detailed explanation the reader is referred to [BCGL] page 206. The notion of distributional problem is crucial to the definition of randomized reductions (which preserve average case complexity). In our system, we assume that R_i and S_i are uniformly distributed in $[0,m)$. This implies that $X_i=R_i+S_i$ has a triangular distribution in [0,2m-1), i.e. $Pr(X_i=x \mid 0 \le x < m)=x/m^2$, and $Pr(X_i=x \mid m \le x < 2m-1)=(2m-x)/m^2$.

In [BCGL] there is also a definition of *randomized Turing reduction* of the kind we need. It has to be efficient, valid, and has to have the *domination* property, which roughly means that the "natural" probability of each instance of C (assuming we reduce B to C) must be $\ge$ the probability to get that instance via a reduction from B, given B's distribution of instances.

4.3 Passive adversary

As mentioned before, the passive adversary is a special case of the malicious adversary, therefore it is sufficient to prove for the latter. However, we believe that reading the proof for the passive adversary helps in understanding the malicious adversary case, therefore we do not omit it. We prove that cracking this system passively, i.e., finding the key $K_{i,j}$ given all the data communicated between the parties is equivalent to breaking the CDH KDS believed to be hard.

The CDH cracking problem (denoted B) is defined as follows: (everything here is *modulo* m, except operations in the exponents which are modulo $\phi(m)$, Euler's totient function, so we won't mention it any more.)

Input: α^x, α^y, α, m **find:** α^{xy} *mod* m

The new cracking problem (denoted C) is:

Input: $X_i=R_i+S_i$, $X_j=R_j+S_j$, α^{S_i}, α^{S_j}, α, m **find:** $\alpha^{R_iR_j} \bmod m$.

We first show that a passive adversary can deduce on the average less than 2 bits of information. This is negligible. Given X_i the adversary may learn something about R_i or S_i, which are supposed to be secret. For example, if $X_i=0$ he can deduce that $S_i=R_i=0$. Likewise, if $X_i=2m-2$ he knows that $R_i=S_i=m-1$. We would like to compute the average number of bits released to the adversary that way. For simplicity we omit in this discussion the subscript i. So we have $x=s+r$. Following the traditional information-theoretic approach we define *equivocation* of a given variable y given x (denoted $H(y\,|x)$) to be the expected amount of freedom of choice of the value of y given x, measured in bits. If $x\in[0,m)$ then the combined uncertainty of r and s is x (r can have any value between 0 and x, but once it is fixed there is no freedom of choice for s). If $x\in[m,2m-1)$ then the combined uncertainty of r and s is $2m-x$ (each of r and s is in the range $[x-m,m)$, which is of size $m-(x-m)=2m-x$). So, we have two triangular functions, which should not be confused. (Here it is the uncertainty function, and previously we discussed the distribution function of X_i). To compute the equivocation we must take the expected value of the logarithm of the uncertainty from 0 to $2m$, but from the symmetry of our uncertainty function and the symmetry of the distribution function of X_i around $x=m$ it follows that it is sufficient to take twice the value from 0 to m. Let $c=\log_2 e\approx 1.44$. We approximate the discrete sum by continuous integral, and get

$$H(r,s\,|x)=2(1/m^2)\int_0^m x\log_2(x)\,dx=2c/m^2(\tfrac{1}{2}x^2(\ln(x)-\tfrac{1}{2})\Big|_0^m=\log_2(m)-c/2.$$

Compared to the maximum possible value of $H(r,s\,|x)$, which is $\log(2m)=\log(m)+1$ we lost less than 2 bits. This is the average number of bits that leak per a single interception. In later section we analyze the average number of bits leaking when r sessions are intercepted, and show it to be of the order of $\log(r)$.

A trivial (worst-case) reduction from B to D can be achieved with $X_i=X_j=0$, $\alpha^{S_i}\equiv\alpha^{-x} \bmod m$, $\alpha^{S_j}\equiv\alpha^{-y} \bmod m$, however, we need a reduction with random X_i and X_j to claim super polynomial security for the new system. (In the trivial reduction the domination property does not hold. All of B's instances are reduced to $X_i=X_j=0$, which is of negligible "natural" probability.)

In problem C, for a given S_i, X_i is uniformly distributed in $[S_i,S_i+m)$. Let D denote the same problem, but we allow X_i to be anywhere in the range $[0,2m-1)$, with the previously mentioned triangular distribution.

In Lemma 1 we show a randomized reduction from B to D. Lemma 2 explains why this gives evidence for the super polynomial security of problem C.

We assume uniform natural distribution for B. That is, α^x and α^y are uniformly distributed in $[0,m)$. Similarly, we assume for problem D that R_i and S_i are uniformly distributed in $[0,m)$. As mentioned earlier, this implies that $X_i=S_i+R_i$ has triangular distribution in $[0,2m-1)$.

$Pr(X_i=x \mid 0 \le x < m) = x/m^2$, and $Pr(X_i=x \mid m \le x < 2m) = (2m-x)/m^2$.

Lemma 1: There exists a randomized Turing reduction from B to D.

Proof: Given an instance of B, create an instance of D as follows: pick random X_i and $X_j \in [0,2m-1)$, with triangular distribution, and set $\alpha^{S_j} \equiv \alpha^{-y} \pmod m$; $\alpha^{S_i} \equiv \alpha^{-x} \pmod m$ (which means $R_i \equiv X_i + x \pmod{\phi(m)}$; $R_j \equiv X_j + y \pmod{\phi(m)}$).

Therefore, the oracle outputs $\alpha^{(X_i+x)(X_j+y)} \equiv \alpha^{X_i X_j} \cdot \alpha^{X_i y} \cdot \alpha^{x X_j} \cdot \alpha^{xy} \pmod m$. The first three multiplicands can be calculated easily, therefore we can also compute the fourth, the desired output of B.

Given the uniform distribution of B's variables, this reduction yields distributions for D's variables, which equals their natural distributions, i.e. the domination property holds. Q.E.D.

Lemma 2: On the average, 2/3 of D's input instances in the construction of lemma 1 are legitimate instances of C.

Proof: For each given S_i, the "legitimate" X_i's are in $[S_i, S_i+m)$, and their probability is the area of the triangular distribution in this interval. We must take the expected value of this probability over all $s \in [0,m)$, where the distribution of $S_i=s$ is uniform in that interval. As before, we use continuous integrals to approximate the discrete sum.

$$E_s Pr(s \le X_i < s+m \mid s) = \int_0^m 1/m \left[\int_s^m (x/m^2)dx + \int_m^{s+m} \left(\frac{2m-x}{m^2}\right)dx \right] ds = 2/3. \quad \text{Q.E.D.}$$

The implication of lemma 2 is that in the reduction of lemma 1, if instead of using oracle D we use oracle C, in 2/3 of the cases the oracle will yield a correct answer. So, we can call oracle C a few times, and use majority voting, to get a negligible probability of error.

This together with the results in [S] and [M] on B imply

Theorem 1: If factorization of RSA modulus is a one-way function with super-polynomial security then the new system has super-polynomial security against passive adversaries.

4.4 Malicious adversary

We apply Definition 3 (malicious adversary) to our system. Suppose the adversary uses some probabilistic poly-time algorithm $h(.)$ on input X_j, i.e., he captures X_j, and instead sends $h(X_j)$ to i. (As before, h may have other inputs like P, S_z, etc. We write it this way just to simplify notations.) When communicating with j, he may act likewise. We do not need the assumption that he uses the same algorithm $h(.)$ in both directions. We'll prove just one way, the other way goes likewise. When i receives $h(X_i)$, she follows the protocol, computing $\underline{K}_{i,j} \equiv (\alpha^{h(x_j)} \cdot \alpha^{-s_j})^{R_i} \pmod m$. We prove now that there is no probabilistic poly-time function $h(.)$, for which the malicious adversary can effectively compute $\underline{K}_{i,j}$.

Assume the contrary. We show a polynomial reduction from B (see section 4.2) to the problem of finding $\underline{K}_{i,j}$, given $X_j, X_i, \alpha^{S_j}, \alpha^{S_i}, \alpha, m$. This reduction is parametrized by h,

i.e., for every given h we give a reduction. We denote this malicious adversary cracking problem, for a given function $h(.)$, by C_h.

The relations between C_h and D_h are the same as between C and D.

To prove that the new system has a super-polynomial security for malicious adversaries we need a randomized reduction from B to D_h, the way we did for passive adversaries.

Lemma 3: There exists a randomized Turing reduction from B to D_h.

Proof: Given an instance α^x, α^y of problem B, we create the following instance of problem D_h: Randomly choose $X_j, X_i \in [0,2m-1)$, with triangular distribution, and set $\alpha^{S_i} \equiv \alpha^{-x} \mod m$; $\alpha^{S_j} \equiv \alpha^{-y} \cdot \alpha^{h(X_j)} \mod m$.

The oracle outputs $\underline{K}_{i,j} \equiv (\alpha^{h(X_j)} \cdot \alpha^y \cdot \alpha^{-h(X_j)})^{R_i} \mod m$, but $R_i \equiv X_i - S_i \equiv X_i + x \mod \phi(m)$, hence $\underline{K}_{i,j} \equiv \alpha^{yx} \cdot \alpha^{y \cdot X_i} \mod m$, but $\alpha^{y \cdot X_i}$ is known, hence so is α^{yx}. Q.E.D.

By analysis similar to that of Lemma 2, and the following remarks, we conclude:

Theorem 2: If factorization of RSA modulus is a one-way function with super-polynomial security then the new system has super-polynomial security against any malicious adversary.

4.5 Amortized Security

We believe that the system leaks on the average the order of $log(r)$ bits when r sessions are intercepted. Clearly this isn't a zero-knowledge system, but a leak of just a few bits buys us much simplicity (compared with [BK]).

Suppose r $X's$ transmitted by A are intercepted. Denote the largest of them X_{max}. Clearly, $X_{max} = S + R_{max}$. Since The $R's$ are uniformly distributed in the interval $[0,m)$, from elementary order statistics we know that the expected value of R_{max} is $m - m/(r+1)$, and the variance of R_{max} is m/r^2. So a reasonable guess for the value of S is just $X_{max} - (m - m/(r+1))$, however, this guess of the value of S has the same variance that R_{max} has, so $S's$ uncertainty losses the order of $log(r)$ bits. Using the other mean values of the $X's$ in a similar way will not further reduce the interval in which S is expected to be.

Acknowledgements

We are most grateful to Stuart Haber for countless helpful discussions. Many thanks are due to Gilles Brassard for crucial remarks concerning lemmas 1 and 3, as well as numerous editorial comments, and to Rich Graveman for many helpful comments regarding the amortized security. Debbie Bloom was helpful in the analysis of the information revealed by the new key-distribution protocol. Finally we thank Pil Lee and Jim Katz for their remarks.

5. Appendix: The definition of security for cryptosystems

We define *security* for cryptosystems using the previous formalism, combined with another measure. We not only assign a probability to each message, but also importance. This measure may be, for example, Shannon's information-content, $-log(p)$, where p is

the probability of the message. We use this particular measure throughout this appendix. The main idea is that when now defining the *cracking rate* of a cryptosystem we compute an entropy function, instead of simple expected value, thus taking into account the "information content" of each cracked message.

While some encryption schemes offer very good protection to every bit of information [GM], it may happen that the same protection value could be achieved with simpler systems, which concentrate on protecting the most important messages. This is captured by our new definition.

A *cryptosystem* (CS) is defined by the following i/o relation:

Input: *clear*: $P=(P_1,P_2)$, *secret*: $U=(U_1,U_2)$, m,

Output: *clear*: $c:=f(P,U,m)$.

As before, P is the set of public keys, U is the set of secret keys. m and c are the message and cryptogram, respectively. f is a probabilistic polynomial time algorithm. For conventional cryptosystems, P is empty, and $U_1=U_2=U$. f is some general function which exists, and therefore we can define i/o relations using it. It is not the actual function used by the parties. For Public-Key systems, the actual function uses just one of the public keys and none of the secret keys, while for conventional systems the actual function has U as a key.

Let $I(\omega,P,c)$ be any probabilistic algorithm trying to compute m. Let $T_I(\omega,P,m)$ be I's running time. Let I's success bit $S_{I,f}(\omega,P,U,m)=1$ *iff* $I(\omega,P,c)=m$ (and 0 otherwise).

Let $g:N\rightarrow N$, be any probabilistic function with ω as one of its inputs. Let d be a distribution function on N, and r a random variable (d's input). The *entropy* of g under the distribution function d is $H_d(g)=-\Sigma_{r,\omega}Pr(d(r))\log(Pr(d(r)))g(\omega,d(r))$. Likewise, d may be a multidimensional distribution function generating P,U,m.

The *Cracking Entropy* for security parameter n is

$CE_{I,f,d}(n)=H_d(S_{I,f}(\omega,P,U,m)/T_I(\omega,P,m))$.

Definition 4: A cryptosystem has at least *security* $s:N\rightarrow N$ if $CE_{I,f,d}(n)=O(1)/s(n)$, for all probabilistic algorithms I.

6. References

[BCGL] Ben-David, S., Chor, B., Goldreich, O., Luby, M.: "On the Theory of Average Case Complexity", *STOC*, 1989 pp. 204-216.

[BK]Bauspiess, F., Knobloch, H.: "How to Keep Authenticity Alive in a Computer Network", *Eurocrypt'89*.

[DEK] Dolev, D.,Even, E., Karp, R.M.: "On the Security of Ping-Pong Protocols", *Information and Control*, Vol. 55, Nos 1-3, Nov. Dec. 1982, pp. 57-68.

[DH] Diffie, W.,, Hellman, M.: "New Directions In Cryptography", *IEEE Trans. on Inf. Theory*, 1976, IT-22, pp. 644-654.

[FS] Fiat, A., Shamir, A.: "How to Prove Yourself: Practical Solutions to Identification and Signature Problems", *Proceedings of Crypto 86.*

[G] Günther, C.G.: "Diffie-Hellman and El-Gamal Protocols With One Single Authentication Key", *Eurocrypt'89.*

[GHY] Galil, Z., Haber, S., Yung, M.: "Minimum-Knowledge Interactive Proofs for Decision Problems", *SIAM J. on Computers* Vol. 18, No. 4, , Aug. 1989.

[GL] Goldreich, O., Levin, A.L.: "A Hard-Core Predicate for All One-Way Functions", *STOC'89* , pp. 25-32.

[GM] Goldwasser, S., Micali, S.,: "Probabilistic Encryption", *JCSS*, Vol. 28, No. 2 , 1984, pp. 270-279.

[GMR] Goldwasser, S., Micali, S., Rackoff, C.: "The knowledge Complexity of Interactive Proof Systems", *Proc. 17th ACM Symposium on Theory of Computing* 1985, and SIAM 1989.

[GMW] Goldreich, O., Micali, S., Wigderson, A.: "How to Play Any Mental Game", *Proc. STOC* 1987, pp 218-229

[HU] Hopcroft,J.E., Ullman, J.D. : *"Introduction to automata theory, languages, & computation"* Addison-Wesley, 1979

[KO] Koyama,K., Ohta, K.: "Identity Based Conference Key Distribution Systems", *Proc. Crypto'87.*

[M] McCurley, K.S.: "A Key Distribution System Equivalent to Factoring", *J. of Cryptology,* Vol.1, No. 2, 1988, pp. 95-106.

[MTI] Matsumoto, T., Takashima, Y., Imai, H.: "On Seeking Smart Public-Key-Distribution Systems", *Trans. of IECE Japan* Vo. E 69, No. 2, Feb 1986.

[O] Okamoto, E.: "Proposal for Identity-Based Key Distribution Systems", *Electronic Letters* 1986, 22, pp. 1283,1284.

[RSA] Rivest, R.L., Shamir, A., and Adelman, L.: "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems", *Commun. ACM* 1978, 21, pp. 120-126.

[S] Shmuely, Z.:"Composite Diffie-Hellman Public-Key Generating Systems Are Hard to Break", TR #356, *Computer Science Dept. Technion, IIT* , Feb. 1985.

[Y] Yacobi, Y.: "Attack on The Koyama-Ohta Identity Based Key-Distribution System", *Proc. Crypto'87* .