TOWARDS PRACTICAL 'PROVEN SECURE' AUTHENTICATED KEY DISTRIBUTION

Yvo Desmedt*
Computer Science Department†
Technion – Israel Institute of Technology
Technion-City, Haifa 32000
Israel

Mike Burmester
Department of Mathematics
RH – University of London
Egham, Surrey TW20 OEX,
U.K.

ABSTRACT

Secure key distribution is a critical component in secure communications. Finding 'proven secure' practical key distribution systems is one of the major goals in cryptography. The Diffie-Hellman variants, a family of key distribution systems, achieve some of the objectives of this goal. In particular, the 'non-paradoxical' system (by Matsumoto-Takashima-Imai and Yacobi) is claimed to be secure against a known-key attack. *In this paper we show that the argument used to prove this is flawed, and we explain how it can be fixed.*

1 INTRODUCTION

The secure distribution of keys is of vital concern to any secure communication. Many key distribution systems have been proposed (*e.g.*, [10, 15, 6, 5, 16, 19, 17, 20]) offering various levels of security and complexity. A lot of research has also focused on the security aspects of such systems and on their weaknesses and strengths (*e.g.*, [21, 23, 22]). Most systems use a public key setting for which a center is required to guarantee the parameters of the system. For practicality, the trust in the center must be kept to a minimum (*e.g.*, restricted to the setting of the system). Practical systems must have a small number of interactions, and a low communication and computation overhead. Ideally, a practical key distribution system should be non-interactive.

A system is 'proven secure' if breaking it is as hard as solving a (believed to be) hard cryptographic 'reference' problem, such as factoring, the discrete logarithm, the Diffie-Hellman problem [10], etc. For practical (conditionally secure) systems this seems to be the best achievable level of security.

The original Diffie-Hellman [10] system is not secure against impersonation attacks. When Alice exchanges a key using Diffie-Hellman, she may think that it is with Bob while in fact it is with Carl. (The public-key version of the Diffie-Hellman scheme always produces the same fixed key, *i.e.*, Alice and Bob will always produce the same session key). This authenticity problem was addressed in [16, 23, 22, 21]. Schemes that solve this problem are often called *authenticated*. Yacobi-Shmuely [23] proposed a family of variants of the Diffie-Hellman system which are provably secure against a ciphertext-only attack by a passive adversary. These systems are very practical: they are non-interactive (what one party sends is independent of what the other sends) and have a low communication and computation complexity. However they [23] are not secure against a known-key attack by a passive adversary, as observed by Yacobi [22]. This means that, if one session key (produced by Alice and Bob using this scheme) is ever revealed, then all (their) further keys can be computed by a passive eavesdropper.

To motivate the necessity for considering our type of active known key attack we consider the following three scenarios:

- *Negotiation of contracts:* During the negotiations for a contract the discussions may have to be private. Once the contract is signed, there is no need to keep secrecy and the keys used to protect privacy may be revealed.
- *Verification of treaties:* A message is authenticated using a conventional cryptosystem. To reduce the possibility of hiding a covert (subliminal) message, the key is revealed immediately after the message is authenticated [1, p. 33].
- *Jealous spouse:* Alice is the jealous spouse who tries to find out either the secret session key of her husband Bob, or the plaintext exchanged with others. She is expecting from Bob a large file, but will not adhere to the rules of the key exchange protocol. In particular she does not know the session key she exchanged with Bob. Then Bob sends her the (encrypted under the session key) file, which of course she cannot decrypt. A few days later she claims to have lost the session key. Bob has no reason not to give it to her.

We conclude that known key attacks in which the key is revealed to an insider almost immediately, are quite realistic.

The 'non-paradoxical'[1] [22, 16] key distribution system seems to overcome the known key attack problem. It is *claimed to be 'proven secure' against a known-key attack by an active impersonator* [22], but the *proof given is flawed*.

The purpose of this paper is to discuss the flaw in Yacobi's argument (Section 3) and to show how it can be fixed (Section 4). In Section 2 we first describe the 'non-paradoxical' system. We conclude with remarks and open problems in Section 5.

*Part of this work was supported by NSF Grants NCR-9106327 and INT-9123464.

†On sabbatical from: Department of EE & CS, Univ. of Wisconsin – Milwaukee, P.O. Box 784, WI 53201 Milwaukee, U.S.A.

1st Conf.- Computer & Comm. Security '93-11/93 -VA,USA
© 1993 ACM 0-89791-629-8/93/0011...$1.50

[1] Although some security aspects of this scheme were proven by Yacobi, Matsumoto-Takashima-Imai first presented the scheme (without any proofs of security).

The term 'non-paradoxical' originates from [14]. Before that paper all proofs of security of 'proven secure' cryptosystems implied that the proof itself could be used to break the cryptosystem using an active adversary (*e.g.*, a chosen text attack). Schemes that overcome this paradox are sometimes called: non-paradoxical.

There are many other key distribution systems which use symmetric keys (*e.g.*, [18]), or for which the security is only heuristic (*e.g.*, [19, 20]). These are not relevant to the problem addressed here, and are not discussed.

2 THE 'NON-PARADOXICAL' KEY DISTRIBUTION SYSTEM

This system [22, 16] uses a composite modulus setting, with p, q appropriate primes, $m = pq$, and α an element of high order in Z_m^*. Each party U_k has a secret key s_k and a public key $y_k = \alpha^{s_k} \bmod m$. To get a session key, users U_i, U_j select random exponents e_i, e_j respectively, and exchange $r_i = \alpha^{e_i} \bmod m$, $r_j = \alpha^{e_j} \bmod m$. U_i computes the session key as $k := r_j^{s_i} \cdot y_j^{e_i} \bmod m$ and U_j as $k := y_i^{e_j} \cdot r_i^{s_j} \bmod m$. Because $k = r_j^{s_i} \cdot r_i^{s_j} \bmod m$, they obtain the same key.

It is shown [22, p. 271] that for this system, the cryptanalytic (cracking) problem under ciphertext-only attack by a passive adversary is as hard the Composite Diffie-Hellman problem $\mathbf{DH_c}$, which is:

> **Input:** $\alpha^x \bmod m$, $\alpha^y \bmod m$, α, m (composite);
> **Output:** $\alpha^{xy} \bmod m$.

The Composite Diffie-Hellman problem is one of the problems believed to be hard. (A problem is hard if it is not possible to compute a solution in probabilistic polynomial time.)
In [22, p. 271] the following argument is given:

> Triples $(r_1' [= \alpha^{e_1} \bmod m], r_2' [= \alpha^{e_2} \bmod m], k' \equiv (\alpha^{s_2})^{e_1} \cdot (\alpha^{s_1})^{e_2} \bmod m)$, can be easily computed, hence they don't contribute any new knowledge ...we address the following disruptive adversary... We can reduce the basic Diffie-Hellman problem to the cracking problem under impersonation attack, with known old sessions' information. Since *old information can be computed by anybody easily*, we can remove this obstacle and concentrate on a reduction to the cracking problem without that history.

One is led to conclude that the cracking problem under any known-key attack, in which the impersonator is allowed to keep a history, is as hard as the $\mathrm{DH_c}$ problem.

3 THE 'NON-PARADOXICAL' SYSTEM LEAKS KNOWLEDGE UNDER KNOWN KEY ATTACK

In Section 3.1 we discuss the flaw informally. For the readers who are familiar with zero-knowledge we give a formal description of the flaw in Section 3.2.

3.1 AN INFORMAL APPROACH

The problem with Yacobi's argument is that it assumes that the adversary $\tilde{U}_i$ will behave in a nice (simulatable) way. We shall see that this need not be the case.

Suppose that U_i is a user who wants at some later stage to impersonate U_j. For this purpose he will gradually accumulate (new) knowledge about the secret key s_j of U_j. His goal is, eventually, to compute s_j, by collating appropriately the knowledge about s_j which he has obtained. For this purpose he will employ the key distribution protocol (U_i, U_j) repeatedly, but will use it in a slightly different way than that formally specified. We shall use the notation $\tilde{U}_i$ to indicate that U_i does not follow strictly the protocol. However we are assuming that $\tilde{U}_i$ knows the secret key s_i.

The strategy of $\tilde{U}_i$ is to run the protocol $(\tilde{U}_i, U_j)$ by using an r_i' for which he *does not* know the discrete logarithm, say e_i'. Since $\tilde{U}_i$ cannot compute the key k', he must abort the protocol. However he will try to get the key k' some other way. The scenario for a known-key attack provides him with a means of (sometimes) obtaining it. For example, U_j may throw the key in a waste paper-basket and $\tilde{U}_i$ pick it out (see also the scenarios considered in the Introduction). Now $\tilde{U}_i$ can calculate $r_i'^{s_j} \equiv k' \cdot r_j^{-s_i} \pmod m$, since he knows s_i. In this way $\tilde{U}_i$ will aquire *new* knowledge about s_j, which he could not have obtained by himself. (For a formal description see Section 3.2. Observe that in the 'honest' case, U_i will get no new knowledge from the key k'. Indeed, in that case U_i would know e_i' and could himself compute $r_i'^{s_j} := y_j^{e_i'} \bmod m$. Of course this knowledge in itself is not sufficient to break the system, but it cannot be discounted as irrelevant. There is no reason to exclude the possibility that such knowledge combined appropriately would, eventually, make it possible for $\tilde{U}_i$ to impersonate U_j. The main point which we want to make is that old information can*not* necessarily be computed easily, so there is a flaw in the original argument.

3.2 A MORE FORMAL APPROACH

In this section we assume that the reader is familiar with the concept of zero-knowledge [13] and zero-knowledge proofs of knowledge [11]. The protocol which $\tilde{U}_i$ uses in the previous section can be described formally as follows:

Input: $\alpha^{s_i} \bmod m$, $\alpha^{s_j} \bmod m$, α, m.
Step 1. $\tilde{U}_i$ selects (using some non-uniform distribution) $r_i' \in Z_m$, and U_j selects e_j, r_j as in the 'non-paradoxical' protocol. $\tilde{U}_i$ sends r_i' to U_j, and U_j sends $r_j = \alpha^{e_j} \bmod m$ to $\tilde{U}_i$.
Step 2. $\tilde{U}_i$ 'obtains' (later) from U_j the key $k' = r_j^{s_i} \cdot r_i'^{s_j} \bmod m$.

Formally, the argument used in the previous section amounts to stating that the 'non-paradoxical' key distribution system under known-key attack is (likely) *not* to be zero-knowledge. Let us be more specific.

Theorem 1 *If the (U_j, U_i) protocol is computationally zero-knowledge, then the Diffie-Hellman problem is easy.*

Proof. (Sketch) Consider a $\tilde{U}_i$ that sends a fixed r_i' (*e.g.*, $2 \bmod m$) without knowing e_i', and assume that α is a generator $\bmod p$ and $\bmod q$. Observe that in the real interaction there is only *one* possible session key k'. From k', s_i and r_j, $\tilde{U}_i$ can easily compute $r_i'^{s_j} := k' \cdot r_j^{-s_i} \bmod m$. So if the protocol were zero-knowledge, then $\tilde{U}_i$ could simulate the *unique* $r_i'^{s_j} \bmod m$ when *only* r_i', $y_j = \alpha^{s_j} \bmod m$, α and m are given (because U_j is honest, r_j can easily be simulated). But this then would imply that the Diffie-Hellman problem is easy. □

So there seems to be no easy way of simulating $k' = r_j^{s_i} \cdot r_i'^{s_j} \bmod m$, unless e_i' is known[2].

It follows that one cannot remove the old session's information and focus on the reduction to the cracking problem *without history* (as in [22, p. 271]). We should point out that we do *not* see how to exploit our attack to cryptanalyze (break) the 'non-paradoxical' key distribution system. So from a heuristic point of view this scheme appears to be as secure as before. Our argument only undermines the proof of its security under an *active* known key attack. It certainly is secure against a *passive* known key attack (as shown in [22]).

[2] Observe that if U_i is honest then the protocol (U_j, U_i) is zero-knowledge *for the honest* U_i. Saying that the protocol (U_j, U_i) is zero-knowledge means that it is zero-knowledge for *all* probabilistic polynomial-time U_i.

4 FIXING THE KEY DISTRIBUTION SYSTEM

To fix the proof we must somehow prevent dishonest parties behaving in the way described above.

The new protocol

Input: $\alpha^{e_i} \bmod m$, $\alpha^{e_j} \bmod m$, α, m.
Step 1. U_i, U_j execute the 'non-paradoxical' key distribution protocol and exchange r_i, r_j, respectively, but they do *not compute the key as yet.*
Step 2. U_i, U_j prove to each other, each using an interactive zero-knowledge proof (*e.g.*, [8, 7, 3]), knowledge of e_i, e_j, respectively, such that $r_i = \alpha^{e_i} \bmod m$, $r_j = \alpha^{e_j} \bmod m$.
Step 3. If U_i accepts the proof of U_j then he computes k; else he halts. Similarly for U_j.

Informally, with this protocol, if $\hat{U}_i$ does not know e_i then he will not succeed in convincing U_j, and the key k will never be computed by U_j, so $\hat{U}_i$ cannot get it in any other way.

Theorem 2 *The cracking problem for the modified 'non-paradoxical' key distribution system under a known-key attack by an (active or passive) adversary is as hard as the Diffie-Hellman problem.*

Proof. **(Sketch)** We use the same argument as in [22], but must show additionally that the 'view' of $\hat{U}_i$ in the protocol $(U_j, \hat{U}_i)$ augmented with the step in which U_j reveals k, can be simulated. We assume that the reader is familiar with the concept of zero-knowledge [13] and in particular with zero-knowledge proofs of knowledge [11].

Suppose that U_j accepts the proof of $\hat{U}_i$. Then with overwhelming probability $\hat{U}_i$ knows e_i' such that $r_i' = \alpha^{e_i'} \bmod m$ (more formally there is a polynomial time probabilistic Turing machine that can extract e_i' from $\hat{U}_i$). It follows that $\hat{U}_i$ induces a probability distribution on the e_i'. Clearly this distribution can be simulated in polynomial time: we just run $\hat{U}_i$ with random independent coin tosses, to produce pairs (e_i', r_i') with the appropriate distribution. The simulation of the honest U_j is trivial and gives (e_j, r_j). Because the proofs given by $\hat{U}_i$, U_j are zero-knowledge, there is no difficulty in simulating them. Now one computes $k' := y_i^{e_j} \cdot y_j^{e_i'} \bmod m$. Observe that the simulation of the view of $\hat{U}_i$ is only statistical zero-knowledge [13], since there is a small (negligible) probability that U_j will accept a $\hat{U}_i$ who does not know e_i'. (Using a similar trick as in [13, pp. 204–205] it can be made perfect zero-knowledge).

There remains a subtle technicality in the sketch. There are two cases: $\hat{U}_i$ does not know s_i, and $\hat{U}_i$ does. The first case is straightforward. In the second case, $\hat{U}_i$ could choose e_i' as a function of s_i. To avoid this technical problem we use a technique similar as in [12] (also used in [2]). We allow the simulator to ask an oracle the discrete log of y_i. □

5 CONCLUSION

We have shown that the 'proof' [22] that the 'non-paradoxical' key distribution system is provably secure against known-key attacks is flawed. Although we have not succeeded in cryptanalyzing the system, it seems unlikely that the unmodified system is provably secure (simulating the appropriate history is as hard as the Diffie-Hellman problem). So the security of the 'non-paradoxical' key distribution system under known-key attacks is *only heuristic.*

We have proposed a modification for which we get provable security under such attacks. However the modified protocol is interactive (whereas the original protocol is non-interactive: the r_i, r_j which the users exchange are independent). This reduces the practicality of the system.

Open problem and questions

- Do there exist *non-interactive* practical 'proven-secure' authenticated key distribution systems? This problem remains open.
- The security of the 'non-paradoxical' key distribution system prevents the cryptanalyst from computing the *whole* session key k (even if earlier session keys are revealed). It does not guarantee that some of the bits of k may leak. So, from a proven secure point of view, if the session key is long then the user will not know which bits to choose.
- A flaw in a formal security proof can have serious security problems. To reduce errors in computer security, formal verification techniques [9], *e.g.*, using automatic theorem provers, are used. Clearly security proofs in the area of communication security (cryptography) must be thoroughly checked. Can automatic theorem provers for such a complicated task be developed?

ADDENDUM

Yacov Yacobi has kindly communicated us his comments. He states:

> Yvo and Mike are right that there is a flaw in my argument, and that some histories may be leaky. So, in general, history cannot be omitted from the definition of B_{kkp}, even for my "non paradoxical" system. When reducing the Composite Diffie-Hellman cracking problem to B_{kkp} of this system, we have to be able to generate a history. This implies that the reduction works only for P-samplable histories (see [4, Def 2, p. 212]). For example, an ordinary execution of the protocol, where the exponents e_i are picked with uniform distribution results a P-samplable history. It seems that in the attack proposed in this paper, if $r_i \equiv \alpha^{e_i} \bmod m$ is picked with uniform distribution in $(1, m)$, the resulting history is still P-samplable. However, if r_i is structured (say has 100 leading zeroes), then, unless the discrete-log problem is computable in probabilistic polynomial time, the resulting history is not P-samplable, and *this* reduction does not exist.
>
> At the time that I wrote the paper [22], I made the implicit assumption that history is "natural." Yvo and Mike are right that other histories should be considered as well, even if they are less plausible. However, a leak of few bits does not by itself imply insecurity, it only implies non zero-knowledge (see [23]).

REFERENCES

[1] ADAM, J. A. Ways to verify the U.S.-Soviet arms pact. *IEEE Spectrum* (February 1988), 30–34.

[2] BEAVER, D., FEIGENBAUM, J., AND SCHOUP, V. Hiding instances in zero-knowledge proof systems. In *Advances in Cryptology — Crypto '90, Proceedings (Lecture Notes in Computer Science 537)* (1991), A. J. Menezes and S. A. Vanstone, Eds., Springer-Verlag, pp. 326–338. Santa Barbara, California, U.S.A., August 11–15.

[3] BELLARE, M., MICALI, S., AND OSTROVSKY, R. Perfect zero-knowledge in constant rounds. In *Proceedings of the twenty second annual ACM Symp. Theory of Computing, STOC* (May 14–16, 1990), pp. 482–493.

[4] BEN-DAVID, S., CHOR, B., GOLDREICH, O., AND LUBY, M. On the theory of average case complexity. In *Proceedings of the twenty first annual ACM Symp. Theory of Computing, STOC* (May 15–17, 1989), pp. 204–216.

[5] BENNETT, C. H., AND BRASSARD, G. Quantum cryptography, and its application to provable secure key expansion, public-key distribution, and coin tossing. In *International Symposium on Information Theory (abstracts)* (September 26–30, 1983), IEEE, p. 91. St. Jovite, Quebec, Canada.

[6] BLOM, R. Key distribution and key management. In *Proc. Eurocrypt 83* (Udine, Italy, March 1983).

[7] CHAUM, D., EVERTSE, J.-H., AND VAN DE GRAAF, J. An improved protocol for demonstrating possession of discrete logarithms and some generalizations. In *Advances in Cryptology — Eurocrypt '87 (Lecture Notes in Computer Science 304)* (1988), D. Chaum and W. L. Price, Eds., Springer-Verlag, Berlin, pp. 127–141. Amsterdam, The Netherlands, April 13–15, 1987.

[8] CHAUM, D., EVERTSE, J.-H., VAN DE GRAAF, J., AND PERALTA, R. Demonstrating possession of a discrete logarithm without revealing it. In *Advances in Cryptology. Proc. of Crypto '86 (Lecture Notes in Computer Science 263)* (1987), A. Odlyzko, Ed., Springer-Verlag, pp. 200–212. Santa Barbara, California, U.S.A., August 11–15.

[9] DENNING, D. E. R. *Cryptography and Data Security*. Addison-Wesley, Reading, MA, 1982.

[10] DIFFIE, W., AND HELLMAN, M. E. New directions in cryptography. *IEEE Trans. Inform. Theory IT-22*, 6 (November 1976), 644–654.

[11] FEIGE, U., FIAT, A., AND SHAMIR, A. Zero knowledge proofs of identity. *Journal of Cryptology 1*, 2 (1988), 77–94.

[12] GALIL, Z., HABER, S., AND YUNG, M. A private interactive test of a Boolean predicate and minimum-knowledge public key cryptosystems. In *Annual Symp. on Foundations of Computer Science (FOCS)* (1985), pp. 360–371.

[13] GOLDWASSER, S., MICALI, S., AND RACKOFF, C. The knowledge complexity of interactive proof systems. *Siam J. Comput. 18*, 1 (February 1989), 186–208.

[14] GOLDWASSER, S., MICALI, S., AND RIVEST, R. A digital signature scheme secure against adaptive chosen-message attacks. *Siam J. Comput. 17*, 2 (April 1988), 281–308.

[15] INGEMARSSON, I., TANG, D. T., AND WONG, C. K. A conference key distribution system. *IEEE Trans. Inform. Theory 28*, 5 (September 1982), 714–720.

[16] MATSUMOTO, T., TAKASHIMA, Y., AND IMAI, H. On seeking smart public key distribution systems. *The Transactions of the IECE of Japan E69*, 2 (February 1986), 99–106.

[17] MCCURLEY, K. S. A key distribution system equivalent to factoring. *Journal of Cryptology 1*, 2 (1988), 95–105.

[18] NEEDHAM, R. M., AND SCHROEDER, M. D. Using encryption for authentication in large networks of computers. *Commun. ACM 21*, 12 (December 1978), 993–999.

[19] OKAMOTO, E. Key distribution systems based on identification information. In *Advances in Cryptology, Proc. of Crypto '87 (Lecture Notes in Computer Science 293)* (1988), C. Pomerance, Ed., Springer-Verlag, pp. 194–202. Santa Barbara, California, U.S.A., August 16–20.

[20] OKAMOTO, E., AND TANAKA, K. Key distribution system based on identification information. *IEEE J. Selected Areas in Commun. 7*, 4 (1989), 481–485.

[21] W. DIFFIE, P. C. V. O., AND WIENER., M. J. Authentication and authenticated key exchanges. *Designs, Codes and Cryptography 2* (1992), 107–125.

[22] YACOBI, Y. A key distribution paradox. In *Advances in Cryptology — Crypto '90, Proceedings (Lecture Notes in Computer Science 537)* (1991), A. J. Menezes and S. A. Vanstone, Eds., Springer-Verlag, pp. 268–273. Santa Barbara, California, U.S.A., August 11–15.

[23] YACOBI, Y., AND SHMUELY, Z. On key distribution systems. In *Advances in Cryptology — Crypto '89, Proceedings (Lecture Notes in Computer Science 435)* (1990), G. Brassard, Ed., Springer-Verlag, pp. 344–355. Santa Barbara, California, U.S.A., August 20–24.