Securing Traceability of Ciphertexts — Towards a Secure Software Key Escrow System*

(Extended Abstract)

Yvo Desmedt**

Department of Electrical Engineering and Computer Science, University of Wisconsin–Milwaukee, WI 53201-0784, U.S.A., e-mail: desmedt@cs.uwm.edu

Abstract. The Law Enforcement Agency Field (LEAF), which in Clipper is appended to the ciphertext, allows the Law Enforcement Agency to *trace* the sender and receiver. To prevent users of Clipper to delete the LEAF, the Clipper decryption box will not decrypt if the correct LEAF is not present. Such a solution requires the implementation to be tamperproof.
In this paper we propose an alternative approach to achieve traceability. Our solution is based on the computational complexity of some well known problems in number theory. So, our scheme does not require a tamperproof implementation, nor a secret algorithm. Its applications extend beyond key escrow.

1 Introduction

Clipper [7] (see also [3, 25]) allows the Law Enforcement Agency to wiretap after having received the proper authorization. Key Escrow Agencies keep shares of the user's secret key which they reveal to the Law Enforcement Agency once the proper authorization has been obtained. To identify the sender and receiver, a field, called the LEAF is attached to the ciphertext. If the LEAF can be removed, then the Law Enforcement Agency cannot request an appropriate court order (since one cannot identify sender or receiver). If the transmission method used is broadcast oriented, such as in cellular telephone, it may be hard to trace the sender and it may even be harder to trace the receiver. In such application it is essential that the functionality of the LEAF is secure.

Let us explain how one guarantees in Clipper that the LEAF is present, *i.e.*, has not been deleted. When a ciphertext is received the correctness of the

* DISCLAIMER: This paper is not intended at all as an endorsement of the Clipper idea or the idea of Key Escrow. The intend of this paper is scientific, *i.e.*, to propose a solution to the open problem whether it is possible to make a software based key escrow system without the need of tamperproofness.

** A part of this work has been supported by NSF Grant NCR-9106327. The author is solely responsible for the content of this paper. A part of this research was done while the author was visiting the Università di Salerno, Italy. This visit was supported in part by CNR AI n.94.00011.

L.C. Guillou and J.-J. Quisquater (Eds.): Advances in Cryptology - EUROCRYPT '95, LNCS 921, pp. 147-157, 1995.

LEAF is checked. If incorrect, the chip refuses[3] to decrypt. So *tamperproofness* is *essential* to the protection method used in *Clipper*.

In this paper we discuss how an alternative to Clipper can be adapted to secure the LEAF. The main goal of the paper is to demonstrate that it may *not* be necessary to rely on tamperproofness to achieve Key Escrow. Our solution is based on computational number theory.

We propose an ElGamal based scheme in which given the ciphertext (and nothing more) it is possible for the Law Enforcement Agency, knowing a trapdoor, to trace the receiver. The trapdoor will not help the Law Enforcement Agency to decrypt ciphertexts. We will argue that any attempt by the sender to hide the identity of the receiver by altering the ciphertext will imply that the receiver can no longer decrypt it properly. We also note that for public key schemes, as RSA, it is very easy for the sender to hide the identity of the receiver when sending ciphertext (see Section 3.1).

To understand the importance of this paper we remind the reader that NIST (National Institute of Standards and Technology, US) has set up a key escrow software working group [27, 28]. The only software solutions presented so far [33] allows any hacker to modify the program so that the ciphertext becomes untraceable. In our approach any software implementation of it seems secure (under reasonable assumptions) against such hacker's attempts.

We note that extensive research has gone towards achieving anonymity and untraceability, *e.g.*, [4, 5, 6], but very little to guarantee traceability against senders and receivers who conspire.

In [9], Threshold Decryption was proposed as an alternative to Clipper. Threshold Decryption allows shareholders to decrypt ciphertext (or a session key) without the need to reveal their shares to a third party. Several Threshold Decryption schemes have been presented. One [10] is based on the ElGamal cryptosystem [14], others [11, 16] are based on RSA [30] and a "proven secure" one is presented in [8]. It is not the purpose of this paper to discuss these ideas in more details. The scheme we present is compatible with the the ElGamal based system Threshold Decryption method, but it is not excluded that this idea would work for other cryptosystems as well.

In Section 3.1 we argue that in existing schemes it is easy to bypass traceability of the sender/receiver of the ciphertext. In Section 3 we discuss the main ideas that we will use to achieve traceability and then discuss how this idea can be set up in an ElGamal setting. Before proposing the actual scheme (see Section 5) we first discuss in Section 4 the privacy of the plaintext. In Section 6 we explain why the traceability of the scheme seems secure.

For the reader who is not familiar with basic algebraic notations and with basic cryptosystems we use, we first overview these.

[3] The security of this check is not very high as was recently demonstrated by Matt Blaze [26]. He demonstrated that, by doing some trial-and-error on the LEAF, Clipper could be used with an incorrect LEAF, implying that the Law Enforcement Agency is unable to eavesdrop. Future implementations of Clipper may have a higher security against such attacks.

2 Background

First we remind the reader how the ElGamal [14] public key encryption scheme works. In the ElGamal scheme, we have a prime p and an element g of large enough order, *e.g.*, a primitive element. Each user publishes as public key $y_j = g^{s_j} \bmod p$, where s_j is the user's secret key (chosen uniformly random in Z_{p-1}). To encrypt the message M $(0 \leq M < p)$ the sender chooses a uniformly random r in Z_{p-1}, which we denote as $r \in_R Z_{p-1}$, (or $r \in_R Z_{\text{ord}\, g}$ if ord g is known) and sends the receiver $(R, C) = (g^r, M \cdot y_j^r) \bmod p$, where y_j is the receiver's public key. Decryption is straightforward. Indeed, the receiver knowing s_j computes $M = C \cdot R^{-s_j} \bmod p$. To simplify notations, we often drop, in the rest of the paper, the $\bmod p$.

We now remind the reader of some basic algebraic [22] notations we will use in this text. Let G be a finite group G and $g \in G$. The subgroup of elements $\{g, g^2, \ldots, g^{d-1}, g^d = 1\}$ is called cyclic and is denoted as $\langle g \rangle$. The smallest d for which $g^d = 1$ is called the order of g and we denote it as: ord(g). The cardinality of a finite group G is called the order of G.

3 Towards the scheme

3.1 Failure of existing schemes

For many cryptosystems the ciphertext itself does not reveal the identity of the sender nor the identity of the receiver. In public key systems the public key is receiver dependent, but that does not mean that an eavesdropper can find the identity of the receiver by observing one ciphertext. In the RSA system the ciphertext is a number between 0 and $n - 1$, but if only one ciphertext block is sent this does not necessarily allow the eavesdropper to find n, which would identify the receiver. Worse, the sender can hide n by using similar techniques as were developed in [16]. Indeed, to hide n, the sender adds random multiples of n to the ciphertext, *i.e.*, $r \cdot n$, where $0 \leq r < \lfloor 2^{2\lceil \log_2 n \rceil}/n \rfloor$. From [16] follows that if plaintext is uniformly distributed (which can roughly be approached using a source coder [18]) then n is well hidden[4]. Traditional schemes based on discrete logarithm have similar problems, as is easy to show.

Above demonstrates that the idea of using a public key scheme is not sufficient to guarantee that one is able to trace the receiver from the ciphertext only.

[4] In fact, the following families of distributions $U(n)$ and $V(n)$ are statistically indistinguishable [21], where $U(n)$ corresponds with choosing uniformly a number between 0 and $2^{2|n|} - 1$ and $V(n)$ corresponds with choosing uniformly a number between 0 and $\left(n \cdot \lfloor 2^{2|n|}/n \rfloor\right) - 1$, as is easy to prove. From this follows that taking polynomially many samples from $U(n)$ and $V(n)$ is indistinguishable.

3.2 Main ideas

To trace the receiver, the main idea is to put redundancy in the ciphertext, which identifies the receiver. The redundancy originates from the public key. The redundancy needs to be added such that if the sender can remove the redundancy, then the receiver can no longer decrypt the ciphertext.

Let us first discuss how this could be realized in general. We continue with our public key scenario. Assume that all ciphertexts sent to whoever all belong to some public set, *e.g.*, Z_p, where p is a prime which has been standardized. To achieve traceability all ciphertexts sent to the receiver, R, should belong to a subset S_R, which is unique for each receiver. To avoid any false identifications, one might require that the subsets are disjoint. So, if given the ciphertext, the Law Enforcement Agency can find (in polynomial time) the subset S_R, then it is able to identify the receiver.

Let us first observe that the requirement that subsets S_R and $S_{R'}$ $(R \neq R')$ be disjoint, is too strong. It is sufficient that the probability of a false identification is very small. One could wonder how one could achieve these subsets. As illustration, we explain this now for the ElGamal scheme.

3.3 ElGamal based subsets

Observe that in the ElGamal scheme $R = g^r \bmod p$ belongs to the group generated by g. To make the ciphertext receiver dependent, R could belong to a receiver dependent subgroup of Z_p^0, where p is a standard prime. To achieve this, let g_j depend on the user j such that $\langle g_j \rangle$ (the cyclic group generated by g_j) is different from $\langle g_k \rangle$ when $j \neq k$.

We now discuss how such $\langle g_j \rangle$ could be obtained. In our first scheme, let m be such that the number of users one can expect is maximum $2^m - 1$. The Law Enforcement Agency chooses different large primes q_i and a prime p, such that $q_i \mid p - 1$. The Law Enforcement Agency also chooses a $g \in Z_p$ of order $Q = \prod_{i=1}^{m} q_i$. The Law Enforcement Agency keeps all q_i secret.

When a user j wants to register a public key, the Law Enforcement Agency chooses a unique binary user identifier $\mathbf{e}_j = (e_1, \ldots, e_m)$, $\mathbf{e}_j \neq \mathbf{0}$, and stores $(j, \mathbf{e}_j)$ in its data base. It then constructs

$$g_j = g^{\frac{Q}{\prod_{i=1}^{m} q_i^{e_i}}} \pmod{p}$$

and gives g_j to the user. The user then proceeds as in ElGamal, it is, chooses a secret $s_j \in_R Z_{p-1}^*$ and constructs $y_j = g_j^{s_j} \bmod p$. The Law Enforcement can easily verify that $y_j \in \langle g_j \rangle$ and that s_j is relatively prime to $p - 1$, by computing $\mathrm{ord}(y_j)$. Indeed, $\mathrm{ord}(y_j) = \mathrm{ord}(g_j)$ iff $y_j = g_j^{s_j}$ and s_j is relatively prime to $p - 1$ [22, p. 45] and [2, p. 89]. Section 3.4 reminds the reader how the Law Enforcement Agency can verify this.

The public key of the user is (g_j, y_j), p is standard. In our first scheme the encryption and decryption will be similar as in ElGamal, except that $r \in_R Z_{p-1}^*$, *i.e.*, relatively prime to $p - 1$.

Observe that there is absolutely no need for the Law Enforcement Agency to reveal $\mathbf{e}_j$ *to user* j. It might be better for the Law Enforcement Agency to keep its data base (i.e., $(j, \mathbf{e}_j)$) secret, which we will motivate in the final paper. Also, it might be better that for all users j the Hamming weight of $\mathbf{e}_j$ is identical, which requires one to make m a little larger (see final paper).

3.4 Tracing a receiver

Let us explain how the Law Enforcement Agency can find the receiver if R is properly constructed. To check this, one first checks whether $R^Q \equiv 1 \bmod p$, if not then $R \notin \langle g \rangle$. Let us assume it is. Observe that the order of g_j is $\prod_{i=1}^{m} q_i^{e_i}$ and that in a finite cyclic group, in this case Z_p^*, the order of a subgroup identifies uniquely the subgroup [22, p. 45]. So to find j it is sufficient for the Law Enforcement Agency to find the order of R modulo p. This can easily be computed. Indeed if $R^{Q/q_1} \equiv 1 \bmod p$ then $e_1 = 0$, else $e_1 = 1$. To find e_i, one continues similarly computing $R^{Q/q_i} \bmod p$. Once the Law Enforcement Agency has found $\mathbf{e}$, it is able to find j using its data base.

In Section 6.2 we will argue (not prove) that the traceability of ciphertext in the scheme is secure against senders who try to fool the Law Enforcement Agency.

3.5 Towards the scheme

The final scheme (Section 5) is almost identical to the one we have discussed in Section 3.3. However, privacy issues, which we discuss in the next section, will force us to slightly modify the scheme.

4 Privacy

Several security aspects must be addressed, but we postpone most of these until we have discussed our scheme in full detail. Here we discuss whether using some g_j which is not a primitive element reduces the security of ElGamal. We also discuss whether the Law Enforcement Agency can decrypt the ciphertext without the help or knowledge of the receiver or the Key Escrow Agencies. We assume in this section that the reader is familiar with basic group theory [22].

First of all we observe that it is not necessary that the plaintext M be in $\langle g_j \rangle$. This implies that, given C, one might be able to compute a part of M. If Q is public, then there exists $M \in Z_p$ such that $M^Q \neq 1 \bmod p$, but $y_j^Q = 1 \bmod p$ for all public keys y_j, so raising $C^Q \equiv M^Q \bmod p$. To analyze the signficance of this we consider the factorization of $p - 1 = q_1^{a_1} \cdots q_m^{a_m} \cdot q_{m+1}^{a_{m+1}} \cdots q_l^{a_l}$. If for all i $(1 \leq i \leq l)$ $a_i = 1$ and $l = m + 1$ and $q_{m+1} = 2$, then the standard ElGamal scheme leaks similar knowledge. (Indeed from the Legendre symbol $(y_j \mid p)$ and $(R \mid p)$ one can compute $(y_j^r \mid p)$ and then $(M \mid p) = (C \mid p) \cdot (y_j^r \mid p)$.)

Since the Law Enforcement Agency knows the factorization it is however able to find much more (without the help of the Key Escrow Agencies) about M than

other eavesdroppers can. Indeed, *for example*, since it knows $\mathrm{ord}(g_j)$ it can compute $C^{\mathrm{ord}(g_j)} \boxminus M^{\mathrm{ord}(g_j)} \bmod p$. Whether such projections really tell something useful about the plaintext is doubtful. However, one could argue that the Law Enforcement Agency (as long as it has not received proper authorization) should not be able to know more about the plaintext than outsiders. We now address this.

5 The scheme

The user's public key is as in Section 3.3, *i.e.*, similar as in the ElGamal scheme with the main exception that g_j is user dependent. Users use this variant of ElGamal *only* to exchange a common session key (so a conventional cryptosystem is used for the actual encryption). It is, the sender chooses a uniformly random $M \in_R Z_p$ and to send it to the receiver, the sender computes $(R, C) = (g^r, M \cdot y_j^r) \bmod p$, where $r \in_R Z^*_{p-1}$. Once M is obtained, sender and receiver hash M to obtain the session key.

The goal of this hash function is to stop the Law Enforcement Agency to learn something extra about the session key. So the hash function should destroy the algebraic properties.

To avoid false identifications in a voluntary[5] escrow system, non-registered receivers should not be allowed to use a $\langle g_j \rangle$ already used by a registered user. This can be achieved when all non-registered users use the *same* g'. This g' is nothing else than a special[6] g_j which would never be assigned to a registered user. If the Law Enforcement Agency identifies an R (of a ciphertext) in $\langle g' \rangle$, then it knows that the ciphertext is sent to a non-registered user. Since a public key system requires an authority (authorities) to manage the public key directory to guarantee authenticity [29], we assume that this authority enforces non-registered user to use g' as the first part of their public key.

In a non-voluntary key escrow system the authority that manages the public key directory will enforce the users to register. This authority could be similar as the telephone monopoly (see also [25]), we therefore call it the monopoly. In this scenario, non-registered users do not have a public key[7], but such users might choose an g_j already in use! To find these violations the monopoly could trace the receiver randomly using non-cryptographic technology and check whether the claimed identity corresponds with the real one (such checks would not affect the privacy of the conversation itself, as we have discussed).

[5] Clipper has been called a "voluntary" key escrow system.

[6] It is important that $\mathrm{ord}(g')$ has no small factor. If it would, then it would allow a "dishonest" sender to expand slightly the bandwidth of the subliminal channel explained in Section 6.2, by having $g_i = g'$. This follows from the fact that ElGamal leaks a little, as explained in Section 4.

[7] They have to use the "public key" system as a conventional cryptosystem or have to rely on non-official authorities that guarantee authenticity of non-registered "public keys". One could compare this with users setting up their own telephone company to hide from the rest of the world.

5.1 Enhancements

To any interested individual, a special office of the Law Enforcement Agency could prove *e.g.*, in zero-knowledge [20], that each prime factor of the order of g is large. If Q and g is public each user can for himself verify that $g_j \in \langle g \rangle$ by checking whether $g_j^Q = 1 \bmod p$. Indeed in a finite cyclic group, in this case Z_p^*, there is at maximum one subgroup of a given order [22, p. 45]. The user should also verify whether $g_j \neq 1 \bmod p$.

To reduce the trust in the Law Enforcement Agency, the concept of mental games (secure distributed computation) [34, 19] can be used allowing several parties to be involved when g and q_i are chosen.

6 Security

We have already addressed the privacy aspect of our scheme. We now wonder whether a collaborating sender and receiver can bypass the traceability. In other words, can a sender send a ciphertext to a receiver using the secret key of the receiver in such a way that the receiver can decrypt it, but the Law Enforcement Agency is mislead, it is identifies the wrong receiver. Before discussing whether we can hide the identity of the receiver, we discuss the model in which we might work.

6.1 Model

We are only concerned about tracing registered receivers (it is receivers whose g_j is constructed by the Law Enforcement Agency). We further restrict ourselves to receivers and senders who never communicated secretly before. Indeed if they ever did, then they could have exchanged a common secret key to be used for any future communication. We also should restrict ourselves to senders and receivers who have no common knowledge besides[8] the registered public keys to be used in the escrow system.

We have to address the question whether a sender can find a method to hide the identity of the receiver. This problem is similar to the subliminal problem introduced by Simmons [32] (for a study in the context of key escrow, consult [23]). In Simmons case two known senders were using an authentication system to hide a secret message, by using a *non-specified* protocol. In our case the users might use a non-specified algorithm to send secret data.

It seems that the model of subliminal-freeness [12] covers this. However, it is immediately clear that if the senders and receivers share a secret key, that

[8] This also excludes public keys used for signatures! Indeed, it has been observed that if a sender knows the receiver's public key for a signature scheme, *e.g.*, DSS [13], then the sender can request the receiver to sign a non-escrowed public key. The sender, after having checked the authenticity of this key, would use it to encrypt data untraceably.

they can send ciphertext which is untraceable (from the point of view of somebody who only has access to the ciphertext and not the physical means of communication). So we must exclude such scenarios. This means in the context of subliminal-freeness that the sender and receiver should not have common knowledge, it is have no knowledge tapes.

6.2 How secure

We were unable to prove that it is impossible to hide the identity of the receiver when sending a ciphertext. Although this might disappoint a theoretician, one should not forget that nevertheless Rivest, Shamir and Adleman *only* argued (and never proved) that their scheme is as secure as factoring, no better attack to invert the RSA function has been found so far.

First let us observe that it seems that the system can be used to send a narrowband subliminal message. Indeed suppose a sender only sends M that contain redundancy. When a sender wants to send a subliminal message to receiver j he uses the public key (g_i, y_i) of *another*[9] receiver i. Clearly j decrypts this ciphertext wrongly and the redundancy will not appear. The mere fact that the receiver cannot properly decrypt is the subliminal information. It is clear that the bandwidth is extremely low. Clearly the Law Enforcement Agency will trace the wrong receiver. One could correctly observe that in our scheme M should be uniform, so above attacks assumes that receiver and sender had some common knowledge, namely the redundancy pattern to check for. However, such common knowledge violates the model.

Let us now restrict our analysis to the case where the receiver (alone[10]) can compute the correct M from the ciphertext and the eavesdropper cannot. Let us assume that the receiver uses ElGamal to decrypt. In this scenario we argue (no proof) that the problem of fooling the Law Enforcement Agency is hard if factoring and discrete log are hard. To succeed, a sender will on input M and (g_j, y_j) compute some (R, C) (not necessarily the specified one) and from (R, C) the receiver can compute the correct M. Then if the receiver (alone) must be able to compute M, one must have that $g_j^{rs_j} = C \cdot M^{-1} \bmod p$. So, given g_j and $y_j = g_j^{s_j} \bmod p$, the sender must compute $R = g_j^r$ and $g_j^{rs_j} \bmod p$. Finding $g_j^{rs_j}$ when given (g_j, R, y_j) is the Diffie-Hellman problem, which is believed to be as hard as finding r (which has been proven for some parameters [24]). So it seems that the sender must know r. To fool the Law Enforcement Agency the sender must manage to compute an R such that $\mathrm{ord}(R) \neq \mathrm{ord}(g_j)$ and $\mathrm{ord}(R) = \mathrm{ord}(g_i)$ for some receiver i. If $R = g_j^r$ this means that $\gcd(r, Q) \neq 1$. So, if the sender must know r then the sender can find a factor of Q. Similar reasonings can be followed for trivial variants of ElGamal.

[9] Another, more successful, variant of the idea of using another person's key to send subliminal data has recently been described [15] (see also [17]). An active party, for example, the phone company, can, however, easily prevent the last attack [15].

[10] In the final paper, we extend the analysis allowing for multiple conspiring receivers.

Finally it seems that knowledge of other g_i will not help a dishonest sender, since finding the order of g_i is believed as hard as factoring $p-1$ [1] and even if g is public finding the discrete log of g_i is believed to be hard.

7 Conclusion

In this paper we proposed an encryption scheme to solve the open problem, posed by NIST, whether a software based approach to Key Escrow is possible.

For our scheme, it seems that if the encryption is done in software the LEAF, being an integral part of the encrypted session key, cannot be removed without destroying ciphertext (the encrypted session key) so badly that it can no longer be decrypted. Another advantage of this scheme is that there is no need to use a tamperproof chip and/or a secret algorithm. The scheme has its full power, enhancing the privacy protection of the user, if Threshold Decryption [10] is used to provide the Law Enforcement Agencies with the decrypted session keys.

The ideas proposed in this paper have broader applications than their use in the context of Clipper only. Indeed these can be used in any setting in which the receiver has to be traceable. We discussed this in the context of Clipper due to its importance at the time this paper was written.

Let us briefly discuss some practical aspects. Let us assume that roughly 10^9 users would use the system then $m = 30$. If each q_i is 320 bits, then Q has 9600 bits, which means that p is only 10 times larger than usually used (we assume that nowadays p of 1024 bits is typical). So our scheme is (less than) 100 times slower than standard ElGamal.

Finally our scheme allows for several enhancements as the use of a verifiable secret sharing scheme [25].

Our paper introduces the following problems. Since the security of our scheme is only heuristic, the open problem is whether one can design a public key encryption scheme in which traceability is proven secure. Also can one make an RSA type variant. Finally can one improve the speed of the scheme.

Acknowledgement

The author thanks Yair Frankel, Sandia National Laboratories, for his comment [15].

References

1. Adleman, L. M., McCurley, K. S.: Open problems in number theoretic complexity. In Discrete Algorithms and Complexity, Proceedings of the Japan-US Joint Seminar (Perspective in Computing series, Vol. 15) (June 4–6, Kyoto, Japan 1986) D. Johnson, T. Nishizeki, A. Nozaki, and H. Wilf, Eds. Academic Press Inc., Orlando, Florida pp. 263–286.
2. Berlekamp, E. R.: Algebraic Coding Theory. Aegen Park Press 1984.

3. Beth, T.: Zur Sicherheit der Informationstechnik. Informatik-Spektrum **13** (1990) 204–215.
4. Chaum, D.: Untraceable electronic mail, return addresses, and digital pseudonyms. Commun. ACM **24** (1981) 84–88.
5. Chaum, D.: Blind signatures for untraceable payments. In Advances in Cryptology. Proc. Crypto'82 (Santa Barbara, 1983) D. Chaum, R. Rivest, and A. T. Sherman, Eds. Plenum Press N. Y. pp. 199–203.
6. Chaum, D.: The dining cryptographers problem: unconditional sender and recipient untraceability. Journal of Cryptology **1** (1988) 65–75.
7. A proposed federal information processing standard for an escrowed encryption standard (EES). Federal Register July 30, 1993.
8. De Santis, A., Desmedt, Y., Frankel, Y., Yung, M.: How to share a function securely. In Proceedings of the twenty-sixth annual ACM Symp. Theory of Computing (STOC) (May 23–25, 1994) pp. 522–533.
9. Desmedt, Y., Frankel, Y., Yung, M.: A scientific statement on the Clipper chip technology and alternatives September 1993. Comment sent to NIST.
10. Desmedt, Y., Frankel, Y.: Threshold cryptosystems. In Advances in Cryptology — Crypto '89, Proceedings (Lecture Notes in Computer Science 435) (1990) G. Brassard, Ed. Springer-Verlag pp. 307–315.
11. Desmedt, Y., Frankel, Y.: Shared generation of authenticators and signatures. In Advances in Cryptology — Crypto '91, Proceedings (Lecture Notes in Computer Science 576) (1992) J. Feigenbaum, Ed. Springer-Verlag pp. 457–469.
12. Desmedt, Y. G.: Subliminal-free cryptosystems. Submitted to the Journal of Cryptology April 1989, revised version submitted May 3, 1994.
13. A proposed federal information processing standard for digital signature standard (DSS). Federal Register August 1991.
14. ElGamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Trans. Inform. Theory **31** (1985) 469–472.
15. Frankel, Y.: February 1995. Personal communication.
16. Frankel, Y., Desmedt, Y.: Parallel reliable threshold multisignature. Tech. Report TR-92-04-02 Dept. of EE & CS, Univ. of Wisconsin-Milwaukee April 1992.
17. Frankel, Y., Yung, M.: Escrowed encryption systems visited: Threats, attacks, analysis and designs. Manuscript, November 1994.
18. Gallager, R. G.: Information Theory and Reliable Communications. John Wiley and Sons New York 1968.
19. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game. In Proceedings of the Nineteenth annual ACM Symp. Theory of Computing, STOC (May 25–27, 1987) pp. 218–229.
20. Goldreich, O., Micali, S., Wigderson, A.: Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. Journal of the ACM **38** (1991) 691–729.
21. Goldwasser, S., Micali, S., Rackoff, C.: The knowledge complexity of interactive proof systems. Siam J. Comput. **18** (1989) 186–208.
22. Jacobson, N.: Basic Algebra I. W. H. Freeman and Company New York 1985.
23. Kilian, J., Leighton, T.: Failsafe key escrow. Tech. rep. Massachusetts Institute of Technology Technical Report MIT/LCS/TR-636 Cambridge, Massachusetts August 1994.
24. Maurer, U. M.: Towards the equivalence of breaking the diffie-hellman protocol and computing discrete logarithms. In Advances in Cryptology — Crypto '94,

Proceedings (Lecture Notes in Computer Science 839) (1994) Y. G. Desmedt, Ed. Springer-Verlag pp. 271–281.

25. Micali, S.: Fair public-key cryptosystems. In Advances in Cryptology — Crypto '92, Proceedings (Lecture Notes in Computer Science 740) (1993) E. F. Brickell, Ed. Springer-Verlag pp. 113–138.
26. Flaw discovered in federal plan for wiretapping. The New York Times June 2, 1994.
27. Opportunity to join a cooperative research and development consortium to develop secure software encryption with integrated cryptographic key escrowing techniques August 24, 1993. NIST.
28. NIST responses to questions from the senate subcommittee on technology and the law, May 3, 1994.
29. Popek, G. J., Kline, C. S.: Encryption and secure computer networks. ACM Computing Surveys **11** (1979) 335–356.
30. Rivest, R. L., Shamir, A., Adleman, L.: A method for obtaining digital signatures and public key cryptosystems. Commun. ACM **21** (1978) 294–299.
31. Schnorr, C. P.: Efficient identification and signatures for smart cards. In Advances in Cryptology — Crypto '89, Proceedings (Lecture Notes in Computer Science 435) (1990) G. Brassard, Ed. Springer-Verlag pp. 239–252.
32. Simmons, G. J.: The prisoners' problem and the subliminal channel. In Advances in Cryptology. Proc. of Crypto 83 (1984) D. Chaum, Ed. Plenum Press N.Y. pp. 51–67.
33. Walker, S. T., Balenson, D. M.: A software key escrow approach, June 10, 1994. Trusted Information Systems, Inc.
34. Yao, A. C.: How to generate and exchange secrets. In 27th Annual Symp. on Foundations of Computer Science (FOCS) (1986) IEEE Computer Society Press pp. 162–167.