æ

A Key Escrow System with Warrant Bounds

Arjen K. Lenstra[1], Peter Winkler[2], Yacov Yacobi[3]

[1] MRE-2Q330, Bellcore, 445 South Street, Morristown, NJ 07960, U. S. A.
E-mail: `lenstra@bellcore.com`
[2] 2D-147, AT&T Bell Laboratories, 600 Mountain Ave., Murray Hill, NJ 07975, U. S. A.
E-mail: `pw@research.att.com`; Research done at Bellcore
[3] MRE-2Q338, Bellcore, 445 South Street, Morristown, NJ 07960, U. S. A.
E-mail: `yacov@bellcore.com`

Abstract. We propose a key escrow system that permits warrants for the interception and decryption of communications for arbitrary time periods, and with either one or two communicating parties specified as the target. The system is simple and practical, and affords reasonable protection against misuse. We argue that use of such a system can produce *both* greater privacy protection and more effective law enforcement than we now enjoy.

1 Background

The tug-of-war between law enforcement agencies and rights advocates regarding communications privacy is not necessarily a zero-sum game. We believe that with a well-designed key escrow system, it is possible to increase the effectiveness of electronic surveillance as an anti-crime measure, while affording better protection to the privacy of citizens.

Although U.S. law enforcement agencies such as the Federal Bureau of Investigation have complained that digital telephony and commercially available cryptography threaten the effectiveness of electronic surveillance, it is a fact that in many respects electronic surveillance is becoming easier.

Electronic surveillance is currently expensive; the average cost of installing and monitoring an intercept in 1993 was $57,256 [1]. There have been only about 900 intercepts ordered per year by state and federal authorities put together, with between 200,000 and 400,000 incriminating conversations recorded annually; the number of *non-incriminating* conversations recorded each year has increased to over 1.7 million. The non-incriminating conversations are weeded out "by hand" at the cost not only of time and money but of the privacy of innocent parties.

Increasingly, however, cordless and cellular telephony permit electronic surveillance without physical access; programmable switches can obviate the necessity for hardware altogether; and digital messaging permits automatic sifting of conversations (by destination, content etc.). Thus the potential exists for cheaper and more effective use of electronice surveillance, and the consequences for the privacy of citizens must be examined carefully.

The availability of public-key cryptography, and the explosion of public awareness in cryptography in general, have put a powerful privacy-enhancing tool into the hands of citizens; conceivably the widespread use of encryption could cripple electronic surveillance as a law enforcement tool. In an effort to provide an alternative, the White House announced on April 16, 1993 the "Escrowed Encryption Initiative." Subsequently the National Institute of Standards and Technology (NIST) approved the *Escrowed Encryption Standard* (EES) for telephone systems [9].

The EES (known often by the name of its chip, "Clipper") caused a substantial outcry [11], partly from cryptologists opposed to the use of a secret algorithm, partly from rights advocates opposed to the whole idea of escrowed keys. The secret algorithm (SKIPJACK) is certainly unnecessary for an escrow system and excellent alternatives have been proposed, e.g., by Micali [8] and Kilian and Leighton [7].

The escrow issue itself is more troublesome. As presently constituted, EES calls for individual keys to be split into the hands of two "trustees" (namely, NIST and a branch of the Treasury Department) who, when served with a lawful warrant, will turn the key over to law enforcement authorities. The warrant itself will contain the usual limitations on target, content and time interval (usually a specified 30-day period), but these limitations do not apply to the key. Instead, the key is supposed to be "returned" (!) at the expiration of the warrant, but non-compliance with this or other Dept. of Justice procedures explicitly "shall not provide the basis for any motion to suppress or other objection to the introduction of electronic surveillance evidence lawfully acquired" [4]. Of course, such a disclaimer is understandable in view of the difficulty of proving (for example) that the FBI no longer has some target's key.

In effect, if citizens a and b give law enforcement authorities reason to believe that they have or will use the telephone to commit a crime, *each* of them gives up his or her "cryptographic rights" for all telephone conversations and for all time—past, present and future. Surely such a concession is unnecessary and excessive, even if one believes that law enforcement authorities have no intention of misusing a key. When automatic sifting of telephone conversations becomes possible, it will be increasingly tempting for the authorities to gather large quantities of data for possible later use when a key is held. But decrypting conversations prior to the start of a warrant would not be very useful since they could not be entered into evidence, and would in fact be illegal to collect.

We believe that a system in which the courts can enforce the terms of a warrant will not only help preserve privacy rights, but will also enable the courts to be more liberal in granting warrants so that federal and state authorities can make better use of electronic surveillance. Micali's system [8] already permits time-bound warrants, for which the law enforcement authorities receive only keys good for the warrant period. We go a step further, allowing the trustees to supply authorities with a key good for only a *pair* of conversants, when appropriate. Thus, for example, if some set of persons is suspected of conspiracy, the courts will be able to issue a warrant for surveillance only of conversations among the targeted parties. Our system will satisfy other requirements as well, detailed below.

2 Requirements

Following are requirements for a key escrow system. We begin with the warrant bounds.

time-boundedness:

It must be possible for the courts to enforce the time-limits of a warrant, by supplying a key that will be effective only for a given period of time (most likely some set of days). As noted above, Micali's system [8] offers a time-bounded option; Clipper of course does not, nor does Kilian-Leighton [7].

target flexibility:

It must be possible for the courts to permit either node surveillance (in which all communications involving a particular target a can be decrypted) or edge surveillance (in which only communications between a and b can be decrypted). None of Clipper, [8] or [7] offers target flexibility.

It is in fact not difficult to design an escrow system that is time-bounded and target-flexible, but we insist that other important characteristics not be sacrificed [4]. In particular, the following properties are desirable and (as we shall see) attainable as well.

non-circumventability:

It should be impossible for a user to *unilaterally* alter his communication protocol in such a way as to obtain encryption from the system without exposing himself to decryption by the proper authorities. Obviously, one cannot stop persons from *colluding* to avoid key escrow, because they can always use their own system (or pieces of the given system); but the system should not provide an obvious way to collude in this manner. (As an example, a system with daily keys must avoid making it easy for persons to cheat by resetting their calendars.)

Roughly speaking, the system of Kilian and Leighton [7] enjoys this non-circumventability, as does Clipper except for the (presumably correctable) flaw found by Blaze [2]. Micali's scheme, however, appears to be unilaterally circumventable when used in a time-bounded manner.

security:

The system should rely on familiar and tested cryptologic techniques. Ideally it should rely on *proven* techniques, but given that there are too few of these, the system should at least avoid techniques that have not

[4] In [5] a "balanced cryptosystem," based on shareable functions, is mentioned where each individual message can be revealed without affecting other messages. It remains to be seen if that cryptosystem applies to our situation, and which of our other properties is satisfied.

built up some empirical credibility. The Micali and the Kilian-Leighton schemes offer reasonable security in this sense and Clipper probably would if the SKIPJACK algorithm were made public; as it is, the public has to rely on a panel report [3].

simplicity:

The system should be practical and understandable; in particular it should not rely on repeated contact between users and trustees, nor should it require many-round preliminaries between communicating parties. It should not offer any impediment to telephone, FAX, or e-mail communication. It should be explicable, in outline if not mathematically, to intelligent lay persons, e.g. the courts.

The Clipper and Kilian-Leighton systems are reasonably simple by this definition; some incarnations of Micali's system are less so, and in particular his time-bounded version requires considerable interaction.

Some additional properties will be discussed later. We now proceed to describe the proposed key escrow system.

3 Preliminaries

Let p and q be two large primes with $q|p-1$, and let $g \in \mathbf{Z}/p\mathbf{Z}$ be an element of order q. For any integer m, we make the obvious identification between $\mathbf{Z}/m\mathbf{Z}$ and $\{0, 1, \ldots, m-1\}$, and between $(\mathbf{Z}/p\mathbf{Z})^*$ and $\{1, 2, \ldots, p-1\}$.

All users of the key escrow system described here share the same p and g. Each user u has a public key $P(u) \in (\mathbf{Z}/p\mathbf{Z})^*$ and a secret key $S(u) \in \mathbf{Z}/q\mathbf{Z}$ such that $g^{S(u)} \equiv P(u) \bmod p$. It is assumed that for all u it is computationally infeasible to derive $S(u)$ from p, g, and $P(u)$—this assumption is based on the supposed difficulty of the discrete logarithm problem. The keys $P(u)$ and $S(u)$ are referred to as the *permanent keys* of user u. The trustees can get their verifiable shares of the permanent keys, with or without thresholds, as described in [10].

Let f be any good conventional block cipher, like (triple) DES. We designate by $c = f(k, m)$ the cryptogram that results from encrypting message m using f with secret key k, and $m = f^{-1}(k, c)$, where c, k, and m are bit strings. It is assumed that m can be derived efficiently from c if and only if k is known, but that k cannot be derived efficiently from c and m.

Furthermore, let $h : \mathbf{Z}/p\mathbf{Z} \times \mathbf{Z}/p\mathbf{Z} \to \mathbf{Z}/p\mathbf{Z}$ be a one way hash function, such that given d and $d_i \neq d$, $y_i = h(x, d_i)$ for a polynomial number of $i \in \mathbf{Z}$ and some unknown x, it is computationally infeasible to find $y = h(x, d)$. The hash function h has to satisfy several other requirements that are specific to our protocol; they will be discussed in the next section.

Both h and f are fixed throughout this paper.

4 The system

We assume that the users of the system agree upon a certain time (presumably UTC), which may for instance be provided by satellite. In what follows we assume that the desired granularity of time for warrants is daily, and we regard any day d as an element of $\mathbf{Z}/p\mathbf{Z}$.

Furthermore we assume that the provider of the communication links (the 'service-provider', e.g. a phone company) is the party that actually provides access to communications for a law enforcement agency who provides the service-provider with a valid warrant. All data obtained by the law enforcement agency from the service-provider should be immediately and unforgeably time-stamped and signed by the latter, so that law enforcement should never be able to pass communication data from one day for data from any other day.

Let $P(a)$, $S(a)$ and $P(b)$, $S(b)$ be the permanent public and secret keys of users a and b, respectively, as described above.

Protocol (for parties a and b on day d):

1. First a and b establish, non-interactively, their session key $k(a,b,d) = k(b,a,d)$, which is computed as $k(a,b,d) = h(P(b)^{S(a)}, d)$ by a and as $k(b,a,d) = h(P(a)^{S(b)}, d)$ by b.
2. Next, before the actual communication using the common key $k(a,b,d)$ takes place, a and b are required to exchange a message to enable law enforcement to compute $k(a,b,d)$ and to decypher the communication between a and b. This is done as follows.
   - 2a. Party a computes $S(a,d) = h(S(a), d)$, and party b computes $S(b,d) = h(S(b), d)$.
   - 2b. Party a computes $S(a,b,d) = h(S(a,d), P(b))$, and party b computes $S(b,a,d) = h(S(b,d), P(a))$.
   - 2c. Party a sends the message $c(a,b,d) = f(S(a,b,d), k(a,b,d))$ to b, and party b sends the message $c(b,a,d) = f(S(b,a,d), k(b,a,d))$ to a.

   (Note that a uses $S(a,b,d)$ as key to encrypt $k(a,b,d)$ using f. Therefore, party b cannot determine the common key $k(a,b,d)$ by decrypting $c(a,b,d)$, but has to compute the common key in Step 1. Similarly a cannot decrypt the message $c(b,a,d)$.)
3. Parties a and b communicate using the conventional block cipher f, with their common key $k(a,b,d)$ as key. All messages encrypted using f and $k(a,b,d)$ are required to have a certain fixed structure, for instance by prefixing them with a certain system dependent header.

Remarks. Because this protocol is non-interactive, it can be used in applications such as e-mail. Note that computing $g^{S(a)S(b)}$ in Step 1 is the greatest part of the computation, and that this value only depends on the communicating parties but not on the day d. This allows users to precompute and store those 'expensive' values for frequent partners, and to do the 'cheap' computations involving h and f on a day to day basis.

Warrants. We now describe the various types of warrants law enforcement might obtain. Unless there is demonstrable cheating by a user, all warrants are time-bounded. In the absence of deliberate collusion, both parties to a conversation must obey the rules of Step 1 in order to create a common session key. This implies that non-collaborative cheating can only take place in Step 2, during which, conceivably, a corrupted c value might be sent. (Note: our description of the warrants assumes that there is a single trustee who knows the secret keys of all users of the escrow system. A threshold secret sharing scheme will be discussed in a later section.)

Possible warrants against a on day d:

1. Edge surveillance: the trustee provides law enforcement with $S(a, b, d)$, for all partners b of a to which the warrant applies.
2. Node surveillance: the trustee provides law enforcement with $S(a, d)$.
3. If a cheats in any part of Step 2: the trustee provides law enforcement with $S(a)$.

In each of these three cases it is clear from the protocol how law enforcement proceeds to obtain the session key(s) for day d. Note that if a cheats in any part of Step 2 and sends a corrupted $\tilde{c}(a, b, d)$ to b, law enforcement will not be able to retrieve the correct session key. In this case, however, law enforcement should be able to convince the trustee that cheating took place, because it is highly unlikely that the ensuing communication between a and b decrypted with the wrong key will have the right fixed structure. Note, however, that the trustee will only provide law enforcement with the user's secret key, if law enforcement can prove, using the time-stamp, that the communication took place on the right day. This leaves the possibility that the user claims a non-malicious error due to noise; we assume that the lower level communication protocols are designed such that this happens with sufficiently low probability.

In principle, the messages in Step 2c do not have to be sent to b and a as long as some party (e.g. the provider of the communication links) records the messages, and forwards them to law enforcement, if appropriate. Sending them to b and a is the most natural solution, however, because the channel between a and b is the channel that will be monitored.

Protection against a frame up. A warrant should not enable anyone to "frame" or "impersonate" any of the parties affected by the warrant. Users are therefore supposed to sign their messages using other systems, with non-escrowed keys, to assure that they cannot be framed by law enforcement or a collusion of trustees. This is necessary in all escrow systems, because anyone with a legal warrant can use the resulting session key to encrypt any message "on behalf" of any user.

Additional requirements on the hash function h. All key escrow systems should be secure against all types of illegal intercepts—also by law enforcement agencies who attempt to exceed their warrant. In this context, known key attacks against key agreement systems are relevant [12].

From the property of the hash function h mentioned in Section 3 it follows that law enforcement cannot predict any past or future session key given only a polynomial number of other session keys. Similarly, any collection of

$S(a,d) = h(S(a),d)$ for some polynomial number of d's in itself is not enough to derive $S(a,\tilde{d})$ for any other previous or future $\tilde{d}$, and neither can a collection of $S(a,b,d) = h(S(a,d),P(b))$ for some polynomial number of b's be used to derive $S(a,\tilde{b},d)$ for any other $\tilde{b}$.

However, the context as well as the values themselves might be known. For example, to protect against finding $S(a)$ given $S(a,d)$, or $S(a,d)$ given $S(a,b,d)$, stronger conditions than are usually considered for hash functions are crucial. Namely, given $S(a,d)$ the secret $S(a)$ is not only a value for which $S(a,d) = h(S(a),d)$, but it is also a value for which the 'correctness' can be verified in another way—by checking whether $h(P(b)^{S(a)},d)$ unlocks the communication in Step 3 in the proper way. And, slightly more involved, given $S(a,b,d)$, the 'node surveillance' key $S(a,d)$ is not only a value for which $S(a,b,d) = h(S(a,d),P(b))$, but also a value for which an $S(a)$ exists for which the correctness can be checked by other means—again by attempting to unlock the communication in Step 3. This is related to so-called 'promise problems,' for which we refer to [6].

Even more complicated conditions have to be imposed on h to make it infeasible to derive useful values from any polynomially sized collection of session keys, $S(a,d)$'s, or $S(a,b,d)$'s. The details are hardly enlightening, however, and we do not elaborate. We note, however, that this is a common problem in applications of hashing functions that is by no means typical for our protocol [13]. Most likely any popular hash function that provides a decent number of bits satisfies all conditions required. Note also that even if different hash functions are used in the different steps of the protocol, we still require that these functions have the more involved properties discussed here.

Security of the protocol. We have not been able to show that our protocol is provably secure or at least as hard to break as some well-established hard-to-solve problem. The following observations might, however, be helpful to shed some light on this matter.

Obviously, for any a and b it should be hard to derive $k(a,b,d) = h(P(b)^{S(a)}, d) = h(P(a)^{S(b)},d)$ from h, d, $P(a)$, and $P(b)$. Otherwise, any eavesdropper would be able to decypher the traffic between a and b, even without having access to $c(a,b,d)$ or $c(b,a,d)$. If h were an invertible one-to-one function, then this cracking problem is at least as hard to solve as the 'Diffie-Hellman problem' (i.e., finding g^{xy} given g^x and g^y). For a one-way hash function h the cracking problem certainly looks even more difficult, in general, but we are not aware of any rigorous results in this direction.

The picture gets considerably more complicated if we consider an eavesdropper who has access to $c(a,b,d)$ or $c(b,a,d)$, or law enforcement who tries to exceed the bounds of a legal warrant. The worst situation, leading to the easiest cracking problem, is where law enforcement tries to exceed the bounds of some number of node surveillance warrants: given $P(a)$, $P(b)$, $k(a,b,d)$, $c(a,b,d)$, $c(b,a,d)$, and $S(a,d)$, for $b \in B$ and $d \in D$, (for some party a, some set B of parties communicating with a, and some set of days D), compute $k(a,b,d')$ for any $b \in B$ and $d' \notin D$.

We require that the hash function h be such that this cracking problem is hard on the average. Note that simple one-wayness or even ordinary 'claw-freeness' may be insufficient because of the dependencies between the various inputs to h and f.

In the next section we discuss the situation where there is more than a single trustee.

5 Threshold secret sharing

To allow more than one trustee in our protocol, we assume that there are $m > 1$ trustees, and that trustee i has a verifiable share $S_i(a)$ of a's permanent secret key $S(a)$, for $1 \leq i \leq m$. Furthermore, we assume that there is some n, $1 < n \leq m$, such that any subset of n trustees can recover $S(a)$ using their $S_i(a)$'s, i.e., we assume the existence of some public function T such that $T(S_{v_1}(a), S_{v_2}(a), \ldots, S_{v_n}(a)) = S(a)$ for any subset $\{v_1, v_2, \ldots, v_n\}$ of $\{1, 2, \ldots, m\}$. The function T is such that any subset of at most $n-1$ trustees has no advantage over anyone else to compute $S(a)$.

If we allow that an n-subset of the trustees, upon receipt of a legal surveillance authorization, computes $S(a)$, and provides law enforcement either with $S(a, d)$, or $S(a, b, d)$, or $S(a)$, depending on the type of authorization and other considerations, then our protocol is unaffected. But for a this situation would be highly undesirable, because, even in the event of the most limited warrant against a, someone would get to see his permanent secret $S(a)$. Therefore, we assume that the trustees are not allowed to collude in this way, and that each trustee is only supposed to provide law enforcement with the trustee's relevant share of information. This requires the following changes in our protocol.

Secret sharing without thresholds. If we only have to allow $n = m$, i.e., if the shares of *all* trustees are required, the change is rather straightforward. Instead of computing $S(a, d)$, party a computes $S_i(a, d) = h(S_i(a), d)$ for $1 \leq i \leq m$ and $S(a, b, d)$ is computed by first computing $S_i(a, b, d) = h(S_i(a, d), P(b))$ and next

$$S(a, b, d) = T(S_1(a, b, d), S_2(a, b, d), \ldots, S_m(a, b, d)),$$

after which $c(a, b, d)$ is computed as usual. The changes for party b are similar.

Depending on the type of warrant, law enforcement either gets the $S_i(a, d)$'s (node surveillance) or the $S_i(a, b, d)$'s (edge surveillance) from all m trustees. In both cases law enforcement can derive the relevant $S(a, b, d)$ or $S(a, b, d)$'s by applying (h and) T. If user a cheats, the $S_i(a)$ are revealed by the trustees, which allows law enforcement to compute $S(a)$ using T.

Secret sharing with thresholds. In the more general case that only the shares of any n-subset of trustees are required, we have to make more extensive changes to our protocol. The solution we present is based on the verifiable threshold secret sharing scheme from [10]. Using this scheme, we may assume that for any subset $V = \{v_1, v_2, \ldots, v_n\}$ of $\{1, 2, \ldots, m\}$ there are non-secret and easily computable constants $b_i(V)$ such that

$$S(a) = \sum_{i=1}^{n} b_i(V) S_{v_i}(a).$$

Without loss of generality we take $\{v_1, v_2, \ldots, v_n\} = \{1, 2, \ldots, n\}$, and we write b_i for $b_i(V)$.

Instead of computing $S(a,d)$ and $S(a,b,d)$ as in Step 2 of the protocol of Section 4, party a computes $S(a,d) = h(d,d)^{S(a)}$ and $S(a,b,d) = h(d,P(b))^{S(a)}$. Furthermore, party a is not only required to send $c(a,b,d) = f(S(a,b,d), k(a,b,d))$ to b, but also an additional message $\bar{c}(a,b,d) = f(S(a,d), k(a,b,d))$. The changes for party b are similar.

For a node surveillance warrant, law enforcement gets $S_i(a,d) = h(d,d)^{S_i(a)}$ for $1 \le i \le n$ (cf. our subset assumption). From these law enforcement can compute $S(a,d) = \prod_{i=1}^{n} S_i(a,d)^{b_i}$, so that $k(a,b,d)$ can be derived from $\bar{c}(a,b,d)$ for any b communicating with a.

For edge surveillance for the communication between parties a and b, law enforcement gets $S_i(a,b,d) = h(d,P(b))^{S_i(a)}$ for $1 \le i \le n$. From these law enforcement can compute $S(a,b,d) = \prod_{i=1}^{n} S_i(a,b,d)^{b_i}$, so that $k(a,b,d)$ can be derived from $c(a,b,d)$.

If user a cheats, the $S_i(a)$ are revealed by n trustees, which allows law enforcement to compute $S(a) = \sum_{i=1}^{n} b_i S_i(a)$.

6 Covert Channels

In [7] it is observed that in many systems, the escrow service can be abused by encoding "shadow" public keys in the data that normally represents ordinary public keys. For example, a long RSA public exponent may represent another public key for a Diffie-Hellman key agreement system. The secret key that corresponds to the shadow public key is never given to the trustees, and thus a covert channel may be established, to which the law enforcement authorities have no access. Yet, the user gets certification services from the system. This form of collusion can be overcome in our system by letting each secret key be a modular sum of two integers, one contributed by the user, the other by the trustees, as noted in [7].

7 Conclusion

A key escrow system is offered that permits cryptographic limits on warrants; in particular it allows the courts to enforce time-limits on surveillance and to target either individuals or pairs of communicating parties. The system is reasonably simple and preserves most of the desirable properties enjoyed by other proposed systems.

Acknowledgments. We gratefully acknowledge valuable suggestions made by Milt Anderson and Ernie Brickell.

References

1. Administrative Office of the United States Courts, 1993, *Report on Applications for Orders Authorizing or Approving the Interception of Wire, Oral, or Electronic Communications (Wiretap Report),* 1993.
2. M. Blaze, "Protocol Failure in the Escrowed Encryption Standard," Proceedings of the 2nd ACM Conference on Computer and Communications Security, November 1994, pp. 59–67.
3. E. Brickell, D. Denning, S. Kent, D. Maher, and W. Tuchman, "SKIPJACK Review: Interim Report, The SKIPJACK Algorithm," July 28, 1993, available electronically from cpsr.org.
4. Department of Justice Briefing re Escrowed Encryption Standard, Department of Commerce, February 4 1994, Washington, DC.
5. A. De Santis, Y. Desmedt, Y. Frankel, and M. Yung, "How to Share a Function Securely," Proc. STOC'94, 1994, pp. 522–533.
6. S. Even, A. Selman, and Y. Yacobi, "The Complexity of Promise Problems with Applications to Public Key Cryptography," *Information and Control*, Vol. 61, No. 2, May 1984.
7. J. Kilian and T. Leighton, "Failsafe Key Escrow," presented at Rump Crypto'94.
8. S. Micali, "Fair public key cryptosystems," Proc. Crypto'92.
9. National Institute of Standards and Technology, *Federal Information Processing Standards Publication 185, Escrowed Encryption Standard*, February 9 1994, Washington, DC.
10. T. P. Pedersen, "Distributed provers with application to undeniable signatures," Proc. Eurocrypt'91, Springer-Verlag LNCS 547, pp 221–238.
11. R. Rivest, "Responses to NIST's Proposal," *Communications of the ACM*, Vol. 35 (7), July 1992, pp. 41–47.
12. Y. Yacobi, "A key distribution "paradox"," Proc. Crypto'90, Springer-Verlag LNCS 537, 1991, pp. 268–273.

This article was processed by the authors using the TEX macro package from Springer-Verlag.