The Subliminal Channel and Digital Signatures

Gustavus J. Simmons

Sandia National Laboratories
Albuquerque, New Mexico 87185

Abstract

In a paper entitled "The Prisoners' Problem and the Subliminal Channel" [1], the present author showed that a message authentication without secrecy channel providing m bits of overt communication and r bits of message authentication could be perverted to allow an $\ell < r$ bit covert channel between the transmitter and a designated receiver at the expense of reducing the message authentication capability to $r-\ell$ bits, without affecting the overt channel. It was also shown that under quite reasonable conditions the detection of even the existence of this covert channel could be made as difficult as the underlying cryptoalgorithm was difficult to "break." In view of this open -- but indetectable -- existence, the covert channel was called the "subliminal" channel. The examples constructed in [1], although adequate to prove the existence of such channels, did not appear to be feasible to extend to interesting communications systems. Fortunately, two digital signature schemes have been proposed since Crypto 83 -- one by Ong-Schnorr-Shamir [2] based on the difficulty of factoring sufficiently large composite numbers and one by Gamal [3] based on the difficulty of taking discrete logarithms with respect to a primitive element in a finite field -- that provide ideal bases for implementing practical subliminal channels. This paper reviews briefly the essential features of the subliminal channel and then discusses implementations in both the Ong-Schnorr-Shamir and Gamal digital signature channels.

* This work was performed at Sandia National Labortories supported by the U.S. Department of Energy under contract No. DE-AC04-76DP00789.

T. Beth, N. Cot, and I. Ingemarsson (Eds.): Advances in Cryptology - EUROCRYPT '84, LNCS 209, pp. 364-378, 1985.

Introduction

The subliminal channel was first conceived of as a way of "cheating" in an authentication without secrecy channel of the type considered for various treaty compliance verification schemes [4,5]. More recently, it has been recognized that several digital signature schemes lend themselves equally well to subliminal communictions. Since there are some (significant?) differences between the two, we briefly review the first formulation -- based on perverting a message authentication without secrecy channel -- and then discuss how such channels can also be concealed in digital signatures.

In order to communicate m bits of information and to provide for r bits of authentication, at least m+r bits must be exchanged. The r bits are in a strict sense redundant information since they are only used by the receiver to partition the set of all possible messages into disjoint subsets of acceptable (i.e., authentic) and unacceptable messages. In complete generality, authentication, with or without secrecy of the information from an opponent depends on the message containing information already known (in some sense) to the receiver. The receiver equates the presence of this prearranged information with the authenticity of the message. Conversely, the absence of this information is interpreted to mean that the communication is not genuine. For example, authentic messages may be required to include a "one time" suffix known in secret to the transmitter and authorized receiver but not to an opponent, as is the common practice in military authentication systems. Since the opponent must be prevented from simply "stripping off" the authenticating information from a genuine message and appending it to a fraudulent or altered message, the information -- both message and authenticating -- is generally secured from outsiders by encryption. In order to make each symbol or collection of symbols in the cipher -- which

the opponent may alter -- be a function of all of the symbols in both the message itself and in the authenticator, the encryption is commonly done as a block cipher (if m+r is small enough) or else as a block chain or feedback cipher, so as to produce the desired "spreading" of symbol dependence. In any event, if the cryptoalgorithm is adequately secure, the probability of the opponent being able to deceive the receiver into accepting a fraudulent or altered message as authentic is bounded by:

$$P_A > 2^{-r} \quad .$$

In a message authentication without secrecy channel, a third party, commonly called the "host" to the communication channel from the origins of this problem in systems to verify compliance with a comprehensive nuclear weapons test ban treaty, is given the means to decrypt the cipher and thus verify that nothing other than the agreed upon message is contained in the cipher. If a single key cryptoalgorithm is used, this is done by giving him the encryption/decryption session key used to encrypt the immediate past message as soon as the exchange has taken place. If a two-key cryptoalgorithm is used, he is given the decryption key in advance of the exchange. For single key cryptographic systems, the host must "trust" the transmitter/receiver until he receives the decryption key corresponding to the last cipher exchange — which if the message is very long may involve an unacceptable level of risk (to him) of covert communication. There is no way of avoiding this problem for single key systems though, since if the host has the key in advance so that he can decrypt the cipher, he could also encrypt and hence create an undetectable forgery. The essential -- and vital -- difference for two key cryptosystems is the absence of this need for even a temporary "trust" by either party of the other since the host can have

the decryption key in his possession prior to any exchange of messages, and hence have the ability to verify the message content prior to forwarding the cipher. On the other hand, since the host cannot infer the unknown encryption key, the transmitter/receiver are confident that he cannot better his guessing odds of choosing an acceptable cipher. Actual authentication without secrecy channels are frequently much more complex than this simplified description suggests. The chapter entitled "Message Authentication Without Secrecy" in Secure Communications and Asymmetric Cryptosystems [4] is recommended for a more complete discussion of this concept.

The essential points to an authentication without secrecy channel are that;

a) the receiver authenticates a message through the presence of r bits of redundant, i.e., expected, information in the decrypted cipher,

b) the host to the communication channel verifies that nothing has been concealed by decrypting the ciphers and verifying that the resulting message is precisely what he expected based on an a priori knowledge of the message.

As mentioned before, the channel is operationally different for the host depending on whether it is based on a single or two key cryptoalgorithm since this determines whether he can check for concealed information before or after the exchange occurs. However, this does not alter the way in which he satisfies himself that nothing is concealed -- namely, that the cipher decrypts to the expected message.

The essential idea involved in setting up a subliminal channel as an indetectable part of a message authentication without secrecy channel is simple. We assume that the authentication channel has been implemented using

a two key cryptoalgorithm. In this case the host and/or opponent are given the decryption key, d, in advance to enable them to verify that the overt channel is not being misused -- which it isn't. The public (decryption) key cryptoalgorithm, though, isn't quite what it appears to be. For the moment, assume that there are two ciphers corresponding to each message, either of which will decrypt, using the public decryption key, into the same (correct) message. The host, given either one of a pair of such ciphers, would decrypt it using his decryption key and be convinced that nothing was hidden in the message which, technically speaking, is true. The receiver however, could in addition to decrypting the cipher to authenticate the message and to recover the overt communication, also be able to learn as much as one additional bit of information from the identity of the particular cipher used to communicate the message. It is this "side" channel that is called the subliminal channel.

Figure 1 shows schematically what the host has agreed to and believes is taking place, i.e., the classical two key message authentication without

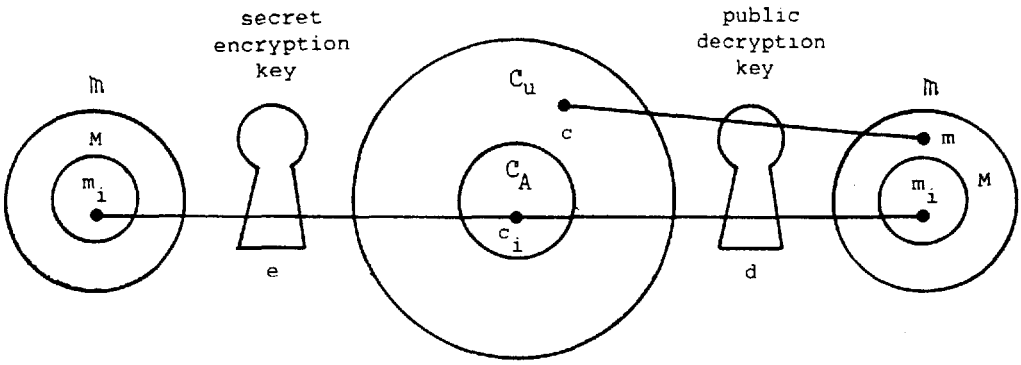

Figure 1. Two Key Message Authentication Without Secrecy Channel

secrecy channel. $\mathbb{M}$ is the set of all possible messages while M is the subset of messages that have the prearranged redundant information and hence will be accepted as authentic by the receiver. For example, if the information is a 48-bit binary number and the authenticating information is a suffix consisting of a 16-bit string of zeroes, $\mathbb{M}$ is the set of 2^{64} 64-bit binary numbers while M is the subset containing only the 2^{48} 64-bit numbers that end in 16 zeroes. It is assumed that the encryption function is a good randomizer, i.e., that the ciphers, C_A, produced by encrypting the messages in M "spread" over the total of 2^{64} ciphers in C in such a way that the opponent -- even if he knows the encryption function (but not the encryption key e of course) and arbitrarily many message/cipher pairs cannot do better at choosing a cipher in C_A than random guessing. The existence of $\mathbb{M}$ as opposed to M is unimportant to the transmitter since he only encrypts messages from the subset M, i.e., messages that will be accceptable to the receiver. The existence of $\mathbb{M}$ is vital, however, to both the opponent and receiver, since it provides the means by which the receiver detects and avoids deception. Using the decryption key, d, the receiver and the host/opponent can decrypt any message in C_A into the proper $m \in M$. C_A is of course unknown (to the opponent) and as difficult to determine as the cryptoalgorithm is cryptosecure. If the opponent chooses a cipher at random, it will with a probability like $\frac{|C_u|}{|C_A \cup C_u|}$ be a cipher in C_u and hence be rejected by the receiver as not being an authentic communication. This is what the host believes is happening, and indeed is all that is verifiable by him.

Figure 2 shows what is actually taking place though (in our simple one-bit example). Instead of there being a single encryption key, e, as claimed by the transmitter/receiver and as believed by the host, there are actually two encryption keys, e_1 and e_2, each of which encrypts the set of acceptable

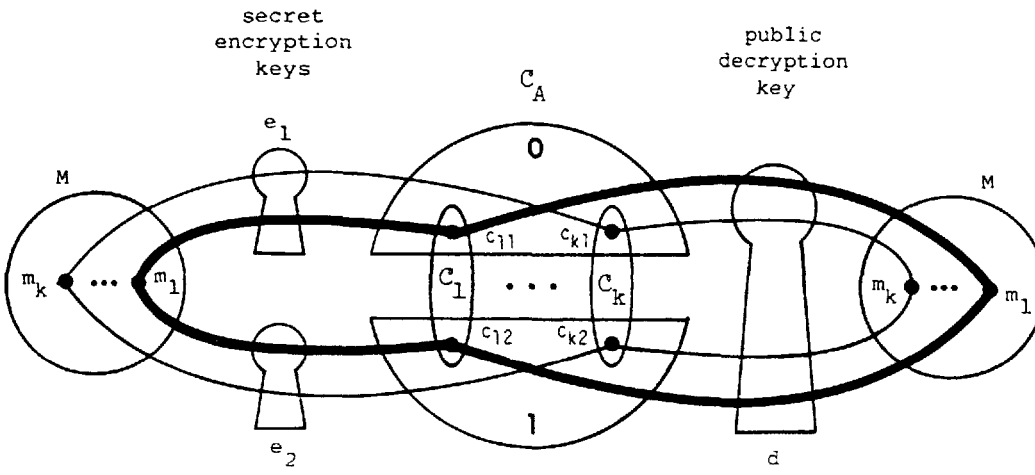

Figure 2. One Bit Subliminal Channel

messages into a corresponding set of acceptable ciphers disjoint from the set of acceptable ciphers produced by the other key. The special feature of the cryptoalgorithm is that either of the ciphers produced by encrypting a message m_i with e_1 or e_2 decrypts under the key d to m_i. As indicated by the bold lines in Figure 2 for the specific choice $m_i = m_1$:

$$E(m_i,e_1) = c_{i1} \neq c_{i2} = E(m_i,e_2)$$

while

$$D(c_{i1},d) = m_i = D(c_{i2},d) \quad .$$

Our convention will be that the transmitter will use e_1 to encrypt if he wishes to send a 0 to the receiver and e_2 to send a 1. The receiver, knowing d, e_1 and e_2 can easily detect the subliminal bit sent by the transmitter. He first decrypts the cipher c using d to recover an $m \in \mathbb{M}$. If the message is authentic, i.e., $m = m_i \in M$ then the received c was actually one of a pair of ciphers, c_{i1} or c_{i2}. If $m \notin M$, then of course the communication would be

rejected by the receiver as inauthentic. If m_i is authentic, he then encrypts it with both e_1 and e_2 to calculate c_{i1} and c_{i2} and hence to determine which cipher was used by the transmitter, i.e., to determine which encryption key was used and thereby to detect the subliminal bit. It should be obvious from this example how the technique can be extended to allow for an arbitrary amount of information to be passed through the subliminal channel. In [1] we discussed one cryptosecure subliminal channel based on the difficulty of factoring sufficiently large products of three distinct primes -- which unfortunately couldn't be extended to practical, large capacity, subliminal channels. In the next section we show how to hide a large capacity subliminal channel in digital signatures.

The Subliminal Channel

Ong, Schnorr and Shamir recently proposed a computationally efficient digital signature channel based on the difficulty of factoring large composite numbers [2]. In the interest of both completeness and brevity we summarize the essential points in their scheme for the three steps: key generation, signature generation and signature verification.

Key Generation

1. Tx chooses a composite n which is computationally infeasible to factor. The factorization of n is kept secret (if known).
2. Tx chooses a random u, (u,n) = 1, and calculates $k \equiv -u^{-2} \pmod{n}$. u is kept secret.
3. Tx publishes n and k as his authentication key.

Signature Generation

Given a message m, (m,n) = 1, to be "signed""

1. Tx chooses a random r, (r,n) = 1. r is kept secret.

2. Tx calculates

$$s_1 = \frac{1}{2}\left(\frac{m}{r} + r\right) \pmod{n}$$

$$s_2 = \frac{u}{2}\left(\frac{m}{r} - r\right) \pmod{n}$$

3. The triple (m, s_1, s_2) is transmitted as the "signed" message.

Authentication of Signature

1. Rx receives (m, s_1, s_2)

2. Rx calculates

$$a \equiv s_1^2 + k \cdot s_2^2 \pmod{n}$$

3. The message m is accepted as authentic if and only if

$$a = m \quad .$$

It is important to note that in the digital signature scheme just described, that if we let $\ell = \lceil \log_2 n \rceil$, 3ℓ bits (on average) are transmitted in a signed message (m, s_1, s_2). This communicates ℓ bits of information overtly, and since there are approximately 2^{ℓ} signatures for any given message, provides approximately ℓ bits of authentication in the signature. The remaining ℓ bits are "wasted" in the digital signature scheme. We propose to use these "free" bits for the subliminal channel. In this respect, using the digital signature channel to implement a subliminal channel differs from what was proposed in [1] where the subliminal bits were obtained by giving up

an equal number of bits from the authentication channel. This difference will also be true for the other digital signature scheme discussed later.

To set up the subliminal channel, in addition to the steps taken by the transmitter in the key generation procedure for the digital signature scheme, the transmitter secretly communicates u to the designated receiver, $Rx^{\dagger}$, for the subliminal channel. Now, when the transmitter wishes to send a signed message m through the overt channel and a covert message m^* through the subliminal channel, where it is still desired that both the $Rx^{\dagger}$ and third parties be able to verify the authenticity of the signature to m, the transmitter generates the signature as follows.

Signature Generation for the Subliminal/Signature Channel

Given a message m, $(m,n) = 1$, to be "signed" and a message m^*, $(m^*,n) = 1$, to be communicated subliminally:

1. Tx calculates

$$s_1 = \frac{1}{2}\left(\frac{m}{m^*} + m^*\right) \pmod{n}$$

$$s_2 = \frac{u}{2}\left(\frac{m}{m^*} - m^*\right) \pmod{n}$$

2. The triple (m, s_1, s_2) is transmitted as the "signed" message.

Authentication of the signature by either the designated receiver, $Rx^{\dagger}$, or by third parties is unaffected by the presence of the subliminal communication. The designated receiver, however, knowing u can solve for the subliminal message as follows:

Decoding the Subliminal Message

The subliminal $Rx^{\dagger}$, given (m, s_1, s_2) and knowing u, calculates

$$m^* = \frac{m}{s_1 + s_2 u^{-1}} \pmod{n}$$

to recover the covert message m^* "hidden" by the Tx in the signature of m.

The 3ℓ bits (on average) contained in the signed message (m, s_1, s_2) have now been fully used to provide for an ℓ bit overt channel, an ℓ bit covert channel and ℓ bits of authentication. An opponent or outsider is faced with an equally difficult (computational) task in detecting either that the subliminal channel exists or is being employed and in breaking the digital signature scheme.

Gamal has proposed a digital signature scheme [3] based on the difficulty of taking discrete logarithms with respect to a primitive element in a finite field GF(p). Following the same procedure adopted in presenting the Ong-Schnorr-Shamir digital signature scheme, the Gamal scheme also involves the same three steps: key generation, signature generation and signature verification.

Key Generation

1. Tx chooses a finite field GF(p), p a prime, and a primitive element $\omega \in GF(p)$. This is public information and need not even be unique to the transmitter.
2. Tx chooses a random u, $u < p$, and calculates $k = \omega^u$. u is kept secret.
3. Tx publishes k -- and if need be p and ω -- as his authentication key.

Signature Generation

Given a message m, $m < p$, to be "signed":

1. Tx chooses a random r, $(r, p-1) = 1$. r is kept secret.
2. Tx calculates

$$x = \omega^r$$

and solves for y in

$$m \equiv ux + ry \pmod{p-1}$$

using the Euclidean algorithm.

3. The triple (m,x,y) is transmitted as the signed message.

Authentication of Signature

1. Rx receives (m,x,y).
2. Rx calculates

$$a = k^x x^y \quad .$$

3. The message m is accepted as authentic if and only if

$$\omega^m = a \quad .$$

In the Gamal digital signature scheme, where $\ell = \lceil \log_2 p \rceil$, just as in the Ong-Schnorr-Shamir digital signature scheme, 3ℓ bits are transmitted to provide an ℓ bit overt channel and ℓ bits of authentication capability. We can use the ℓ bits left over to achieve another subliminal channel.

To set up the subliminal channel, in addition to the steps taken by the transmitter in the key generation procedure, the transmitter secretly communicates u to the designated receiver, $Rx^\dagger$, for the subliminal channel. Now, when the transmitter wishes to send a signed message m through the overt channel and a covert m* through the subliminal channel -- where it is

still desired that both the $Rx^{\dagger}$ and third parties be able to verify the authenticity of the signature to m, the transmitter generates the signature as follows:

Signature Generation for the Subliminal/Signature Channel

Given a message m, $m < p$, to be "signed," and a message m^*, $m^* < p$, to be communicated subliminally:

1. Tx calculates

$$x = \omega^{m^*}$$

and solves for y in

$$m = ux + m^*y \pmod{p-1}$$

using the Euclidean algorithm.

2. The triple (m,x,y) is transmitted as the signed message.

Authentication of the signature by either the designated receiver, $Rx^{\dagger}$, or by third parties is unaffected by the presence of the subliminal communication. The designated receiver, however, knowing u can solve for the subliminal message as follows:

Decoding the Subliminal Message

The subliminal $Rx^{\dagger}$, given (m,x,y) and knowing u, calculates

$$m^* = y^{-1}(m-ux) \pmod{p-1}$$

to recover the covert message m^* "hidden" by the Tx in the signature of m.

The general principles underlying the implementation of a subliminal channel in a digital signature scheme, as illustrated in the preceding two

examples, are probably applicable to digital signature schemes in general. One of the author's colleagues, John DeLaurentis, has shown how to realize a subliminal channel in the earlier Ong-Schnorr digital signature scheme [6] and the author has more recently shown how to use the cubic OSS-signature scheme [7] in a similar manner. Both of these cases are more complex to use than the two discussed here -- but are fundamentally the same. The bottom line is that (several) digital signature schemes can be adapted to provide high capacity subliminal channels -- in which equally much information flows through the covert channel as through the overt channel.

Postscript

In the week following Eurocrypt 84 at which this paper was presented, J. M. Pollard successfully cryptanalyzed the Ong-Schnorr-Shamir digital signature scheme [8]. This development doesn't affect the validity of the concept of the subliminal channel, but it does eliminate from consideration what was the most attractive and practical implementation.

References

1. G. J. Simmons, "The Prisoners' Problem and the Subliminal Channel," Proceedings of Crypto 83, Santa Barbara, CA, August 21-24, 1983, to be published by Plenum Press.

2. H. Ong, C. P. Schnorr and A. Shamir, "An Efficient Signature Scheme Based on Quadratic Equations," to appear Proceedings of 16th Symposium on Theory of Computing, Washington D.C., April 1984.

3. T. El Gamal, "A New Public Key Cryptosystem and Signature Scheme Based on Discrete Logarithms," to appear IEEE Transactions on Information Theory.

4. G. J. Simmons, "Message Authentication Without Secrecy," in Secure Communications and Asymmetric Cryptosystems, ed. by G. J. Simmons, AAAS Selected Symposia Series, Westview Press, Boulder, CO (1982), pp. 105-139.

5. G. J. Simmons, "Verification of Treaty Compliance -- Revisited," Proceedings of the 1982 Symposium on Security and Privacy, Oakland, CA, April 25-27, 1983, pp. 61-66.

6. H. Ong and C. P. Schnorr, "Signatures through Approximate Representations by Quadratic Forms," Proceedings of Crypto 83, Santa Barbara, CA, August 21-24, 1983, to be published by Plenum Press.

7. C. P. Schnorr, "A Cubic OSS-Signature Scheme," private communication, May 1984.

8. J. Pollard, "Solution of $x^2 + ky^2 = m \pmod{n}$," private communication, April 1984.