AUTHENTICATING MULTI-CAST INTERNET ELECTRONIC MAIL MESSAGES USING A BIDIRECTIONAL MAC IS INSECURE

Chris J. Mitchell

Computer Science Department
Royal Holloway and Bedford New College
University of London
Egham Hill
Egham
Surrey TW20 0EX
England

21st May 1990

ABSTRACT

The 1988 version of the message encryption and authentication procedures for Internet electronic mail makes use of a 'Bidirectional MAC' or BMAC. When used for multi-cast electronic mail it is important that this BMAC acts as a one-way function. We show here that it is not a one-way function, which means that the BMAC technique should not be used for authenticating multi-cast messages.

I. INTRODUCTION

The 'Internet' electronic mail system is widely used both within the U.S. and world-wide. Because of the practical importance of this electronic mail network, and because of the sensitive nature of some of the information sent over the network, a 'Privacy Task Force' was set up to recommend ways in which the system could be made secure.

As a result of their efforts, a number of 'Request For Comments' (RFC) documents have been produced: RFC989, [10], in February 1987, RFC1040, [11], in January 1988 (a revised version of RFC989) and, most recently, RFCs 1113, [12], 1114, [9] and 1115, [13], in August 1989 (superseding RFC 1040). These documents describe procedures for message encipherment and authentication.

Unfortunately, the authentication scheme described in RFC989 contains a serious flaw when used for multi-cast messages (i.e. messages sent to more than one recipient). This flaw enables one recipient of such a message to successfully send a bogus message to another recipient in an undetectable way, [15]. We now briefly consider the nature of this flaw, described in more detail by Mitchell and Walker, [16].

The scheme described in RFC989 relies on the sender of a message computing a 'digest' of the message using a pre-determined 'hash function', h say. The digest is then encrypted using each of the recipient keys in turn, and all

the encrypted digests are sent with the message. These recipient keys are known only to the message originator and the particular recipient. The effectiveness of the scheme depends on the hash function *h*, which must satisfy a number of properties, and the encryption of the digests (this latter point we do not consider here). The first important property *h* must satisfy is that if the message *M* is input to *h*, then the output *h*(*M*) (typically of 64 or 128 bits) must depend on all of *M*. The second requirement is that *h* should exhibit the following 'one way' property:

> **H1.** Given any possible candidate for a digest, *C* say, it must be computationally infeasible to find a message *M* such that *h*(*M*) = *C*.

In the scheme described in RFC989, [10], it is suggested that the U.S. standard DES algorithm, [1], [6] is used in the standardised Cipher Block Chaining Mode, [2], [7], [8] to produce a 64-bit Message Authentication Code or MAC. This is precisely the technique standardised in the U.S. for authenticating financial messages, [3], [4]. (Note that the description of CBC mode given in Mitchell, [15], is incorrect).

The key used to compute this MAC is different for each message, and is sent (along with the digest) encrypted under each of the recipient keys. This means that, for the system to be secure for multi-cast messages, the DES CBC MAC must satisfy **H1** even when the key is known. Unfortunately this is

not the case, [15], and this is the source of the weakness in RFC989.

As a result, RFC1040 ([11], section A.2) describes a second function for computing digests called a 'Bidirectional MAC' or BMAC, and suggests that this function should be used for multi-cast mail instead of the conventional MAC. Presumably it was hoped by the authors of RFC1040 that the BMAC function satisfies **H1** even when the key used to generate it is known; unfortunately this is not the case, as we describe in the next section. Showing that BMAC does not satisfy H1 is the main purpose of this short paper.

Perhaps because of the attack described below, the BMAC algorithm has been dropped from the latest, August 1989, set of RFCs, [9], [12], [13]. The algorithm which replaces it, described in RFC 1115, [13], appears far stronger.

II. THE BMAC IS NOT A 1-WAY FUNCTION

A. Definition of the BMAC

We start by describing how a BMAC is computed. To do this we first describe how a MAC is computed. The message to be processed is first divided into a sequence of 64-bit blocks (if necessary, the last block is padded out):

$$M_1, M_2, \ldots, M_r$$

say. The sequence

$$C_1, C_2, \ldots, C_r$$

is then computed, where

$$C_i = E_K\{ M_i + C_{i-1} \} \qquad (1)$$

where, by convention, C_0 is the all-zero block and, as throughout this paper, + denotes bit-wise exclusive-or and $E_K\{\}$ denotes DES encryption using the key K. The MAC is then equal to the final block C_r.

To compute the BMAC, the message is again divided into the sequence (M_i) $(1 \le i \le r)$. The sequence (C_i) $(1 \le i \le r)$ is computed as before, and, in addition the sequence (B_i) $(1 \le i \le r)$ is computed, where

$$B_i = E_K\{ M_i + B_{i+1} \} \qquad (2)$$

where, by convention, B_{r+1} is the all-zero block. The BMAC then consists of the pair of 64-bit blocks (C_r, B_1). In other words, the BMAC consists of a pair of MACs, one computed forwards and the other computed backwards.

B. Breaking the BMAC

We now show how, given any pair of 64-bit blocks and a key K, a message can be constructed which gives this pair of blocks as its BMAC. This implies that the BMAC function is not one-way and hence the RFC1040 scheme is not secure.

The method we propose here allows us to choose the message arbitrarily, except that it must contain two 64-bit blocks which will be essentially random in nature. These 'garbage' blocks are present in order to make the BMAC come out right. In the method described below, these two blocks will be at the beginning and at the end of the message; however, the method can be very simply modified to allow them to be located almost anywhere in the message.

Suppose that the supplied pair of blocks is (C,B) and the supplied key is K. Suppose also that the message to be 'matched' to the BMAC (C,B) is M, where M can be divided into two parts M_1, M_2. In addition choose at random a 64-bit block X. We now do two sets of very similar computations.

First prepare 2^{32} variants of M_1, each variant consisting of a whole number of 64-bit blocks (although the number of blocks in each variant may vary). Davies and Price ([5], pages 278, 279) illustrate a simple technique by which this may be done so that each variant is valid English and each variant is semantically the same. Basically, if n points are identified within the message at which two possible wordings have the

same meaning, then 2^n different variants of the message may be derived. One simple possibility is to insert either one or two spaces after the end of each sentence.

For each variant do the following computations (where we suppose that N_2, N_3, ..., N_s is the decomposition of the variant into 64-bit blocks - note that the first block of the variant is not defined at this stage):

- For every i (i = s, s-1, ..., 2) let:

$$F_i = E_K\{ N_i + F_{i+1} \} \quad (3)$$

  where $F_{s+1} = X$.

- Let:

$$N_1 = D_K\{ B \} + F_2 \quad (4)$$

  where $D_K\{\}$ denotes DES decryption using the key K. N_1 then constitutes the first block of this variant of M_1.

- For every i (i = 1, 2, ..., s) let:

$$G_i = E_K\{ N_i + G_{i-1} \} \quad (5)$$

  where G_0 is the all-zero block.

- Finally let:

$$Y = G_s \quad (6)$$

Each variant (N_1,...,N_s) is then stored along with its corresponding value of Y.

Second prepare 2^{32} variants of M_2, each variant consisting of a whole number of 64-bit blocks (although the number of blocks in each variant may vary). For each variant (having sequence

of 64-bit blocks P_{S+1}, P_{S+2}, ..., P_{r-1}, say) do the following computations:

- For every i ($i = s+2, s+3, ..., r$) let:

$$H_i = D_K\{ H_{i-1} \} + P_{i-1} \quad (7)$$

  where $H_{S+1} = X$.

- Let:

$$P_r = D_K\{ H_r \} \quad (8)$$

  where P_r then constitutes the last block of this variant.

- For every i ($i = r-1, r-2, ..., s$) let:

$$L_i = D_K\{ L_{i+1} \} + P_{i+1} \quad (9)$$

  where $L_r = C$.

- Finally let:

$$Z = L_s \quad (10)$$

This value of Z is then compared with all the values of Y resulting from the 2^{32} variants of M_1. There is a good chance that, before all 2^{32} variants of M_2 have been processed, a match (i.e. a pair of values Y, Z such that $Y = Z$) will be found; we justify this claim below. Now suppose that the sequences (N_i) $(1 \leq i \leq s)$ and (P_j) $(s+1 \leq j \leq r)$ give a match (i.e. using the above notation they have $L_S = G_S$).

We now show that the sequence of blocks obtained by concatenating these two sequences, i.e. the sequence

$$N_1, N_2, ..., N_S, P_{S+1}, P_{S+2}, ..., P_r$$

will have BMAC (C,B).

First let the sequences (C_i) and (B_i) $(1 \leq i \leq r)$ be defined from this concatenated sequence as in (1) and (2) above. We need to show that $C_r = C$ and $B_1 = B$.

Comparing equations (1) and (5) we see that

$C_i = G_i$ $(1 \leq i \leq s)$.

Now, since $Y = G_s$ (by (6)), $Y = Z$, and $Z = L_s$ (by (10)), if we invert (9) we obtain:

$C_i = L_i$ $(s \leq i \leq r)$

and, since $L_r = C$, we have $C_r = C$.

Similarly, comparing equations (2), (7) and (8) we see that

$B_i = H_i$ $(s+1 \leq i \leq r)$.

Now, since $X = H_{s+1}$ (by (7)) and $X = F_{s+1}$ (by (3)), (3) immediately yields:

$B_i = F_i$ $(2 \leq i \leq s+1)$.

Finally, since $B_1 = E_K\{ N_1 + B_2 \}$, (4) immediately gives us $B = B_1$, as required.

C. Remarks on the above attack

The above attack does require a non-trivial amount of processing time and data storage, although neither of these two requirements make the attack infeasible. The attack requires the processing of some 2^{33} part-messages, and a number of DES encryptions/decryptions are required for each

such part-message. However, the key is fixed throughout, and DES hardware is both available and cheap which can perform in excess of 10^5 DES encryptions/decryptions per second (i.e. in excess of 2^{33} DES operations in a 24-hour period). Even in software, DES can be made to run reasonably fast; implementations exist capable of performing 10^4 DES operations per second on a personal work-station (given the key is fixed). Thus if we assume that each part-message contains 10 DES blocks (i.e. 640 bytes), the DES processing could be completed in 80 workstation-days. Such a resource would often be trivially available (using 'spare' machine cycles) to anyone working in, say, an academic computing envorinment.

The storage requirement for the above method is probably slightly more difficult to achieve, although it is by no means infeasible to a determined attacker. It requires the storage of 2^{32} message variants, together with their corresponding values of Y. Indeed, it would be reasonable to sort these variants by their values of Y to make the matching process more simple. This is a lot of data, namely $32(m+1)$ Gigabytes, where m is the number of blocks in each variant. For example, if we let m be 10 (as before) we require a data store of 350 Gbytes (and the processing time for sorting this data). On-line stores of this size, although very expensive to purchase, commonly exist in large commercial and academic institutions, and will probably become relatively inexpensive within the next few years.

Finally note that we asserted in the above text that the probability of a match being found was good, without giving any justification for such a claim. In general, if samples of size u and v are drawn independently at random from a population of size N with replacement, then the probability that there will be a match between the two samples is approximated by

$$P_{\text{match}} = 1 - ((N-u) / N)^v = 1 - (1 - u/N)^v.$$

given u is small compared to N. Under the same assumption, a further approximation gives:

$$P_{\text{match}} = 1 - \exp(-uv/N).$$

If $N = 2^{64}$ and $u = v = 2^{32}$, we get $P_{\text{match}} = 1 - 1/e > 0.5$, thus justifying our claim (if we assume that the DES encryption operation generates randomly distributed 64-bit blocks).

The assumption $u = v = 2^{32}$ was chosen primarily for illustrative purposes, and could, for example, be adjusted so as to give a higher probability of success. Alternatively the values could be adjusted to reflect the available resources of processing and/or storage. For example, u (the number of variants of M_1) might be chosen to be significantly smaller, and v correspondingly higher, so as to reduce the storage requirements (at the cost of increasing the DES processing requirement).

III. RECOMMENDATIONS

In any authentication scheme for multi-cast messages using the method described in the Internet RFCs, it is most important that the BMAC scheme should not be used. As far as alternatives are concerned, the basic recommendation given by Mitchell and Walker, [16], remains sound. That is, the function *h* should be chosen to have the property **H1**.

A suggestion for how such a function can be constructed from a block cipher such as DES has been put forward by Winternitz, [16], [17], [18], and this method remains a good candidate for schemes such as the Internet mail system. Alternative methods based on the use of DES are discussed by Merkle, [14]. Other promising techniques have recently been proposed by Rivest, including the algorithm to be found in RFC 1115, [13], and an even more recent technique known as MD4.

REFERENCES

[1] ANSI X3.92-1981, *Data encryption algorithm*, American National Standards Institute (New York), 1981.

[2] ANSI X3.106-1983, *American National Standard for Information Systems - Data Encryption Algorithm - Modes of Operation*, American National Standards Institute, New York, 1983.

[3] ANSI X9.9, *Financial institution message authentication (wholesale)*, American Bankers Association, Washington, DC, August 1986.

[4] ANSI X9.19, *Financial institution retail message authentication*, American Bankers Association, Washington, DC.

[5] D.W. Davies and W.L. Price, *Security for computer networks*, Chichester, U.K.: John Wiley and Sons, 1984.

[6] FIPS PUB 46, *Data encryption standard*, Federal Information Processing Standards Publication 46, National Bureau of Standards, U.S. Department of Commerce, Washington, DC, January 1977.

[7] FIPS PUB 81, *DES modes of operation*, Federal Information Processing Standards Publication 81, National Bureau of Standards, U.S. Department of Commerce, Washington, DC, December 1980.

[8] ISO 8372, *Information processing - Modes of operation for a 64-bit block cipher algorithm*, International Organization for Standardization, 1987.

[9] S.T. Kent and J. Linn, "Privacy enhancement for Internet Electronic Mail: Part II - Certificate-based key management [Draft]," *Request for Comments 1114 (RFC 1114)*, IAB Internet Privacy Task Force, August 1989.

[10] J. Linn, "Privacy enhancement for Internet Electronic Mail: Part I - Message encipherment and authentication procedures," *Request for Comments 989 (RFC 989)*, IAB Internet Privacy Task Force, February 1987.

[11] J. Linn, "Privacy enhancement for Internet Electronic Mail: Part I - Message encipherment and authentication procedures," *Request for Comments 1040 (RFC 1040)*, IAB Internet Privacy Task Force, January 1988.

[12] J. Linn, "Privacy enhancement for Internet Electronic Mail: Part I - Message encipherment and authentication procedures [Draft]," *Request for Comments 1113 (RFC 1113)*, IAB Internet Privacy Task Force, August 1989.

[13] J. Linn, "Privacy enhancement for Internet Electronic Mail: Part III - Algorithms, modes and identifiers [Draft]," *Request for Comments 1115 (RFC 1115)*, IAB Internet Privacy Task Force, August 1989.

[14] R.C. Merkle, "One way hash functions and DES," Preprint (Xerox PARC), 1989.

[15] C.J. Mitchell, "Multi-destination secure electronic mail," *The Computer Journal*, vol. 32, pp.13-15, 1989.

[16] C.J. Mitchell and M. Walker, "Solutions to the multidestination secure electronic mail problem," *Computers and Security*, vol. 7, pp.483-488, 1988.

[17] R.S. Winternitz, "Producing a one-way hash function from DES," *Advances in Cryptology: Proceedings of Crypto 83*, Plenum, New York, 1984, pp.203-207.

[18] R.S. Winternitz, "A secure one-way hash function built from DES," *Proceedings of the 1984 IEEE Symposium on SEcurity and Privacy*, IEEE, 1984, pp.88-90.