A Chosen Ciphertext Attack Against Several E-Mail Encryption Protocols

Jonathan Katz[*] Bruce Schneier[†]

June 23, 2000

Abstract

Several security protocols (PGP, PEM, MOSS, S/MIME, PKCS#7, CMS, etc.) have been developed to provide confidentiality and authentication of electronic mail. These protocols are widely used and trusted for private communication over the Internet. We point out a potentially serious security hole in these protocols: any encrypted message can be decrypted using a one-message, adaptive chosen-ciphertext attack. Although such attacks have been formalized mainly for theoretical interest, we argue that they are feasible in the networked systems in which these e-mail protocols are used.

1 Introduction

Electronic mail (e-mail) has become an essential communication tool. The ease of e-mail communication, when compared to traditional choices such as physical mail, fax, or telephone, makes it the communications medium of choice for many people. As more people and businesses move on-line, and Internet access becomes more commonplace, e-mail is expected to become even more important as a communications tool.

In order for e-mail to fully supplant other alternatives, businesses (and, to a lesser extent, individual users) must be assured of the privacy of their e-mail correspondence. This is of special concern in the case of e-mail, since it is no doubt easier for an adversary to "tap" into an Internet link than to tap into a phone line. Furthermore, e-mail has the potential of offering security beyond that of telephone conversations, as there is currently no good way of "scrambling" a telephone conversation (although one can detect—and with a bit more trouble, prevent—the "tapping" of the phone line in the first place). It is primarily for these reasons that e-mail encryption protocols were developed.

In any cryptosystem, there are multiple points which an adversary can attack; of course, a cryptosystem is only as strong as its weakest point of attack. In this paper, we point out a so-called *chosen ciphertext* attack which succeeds against all current implementations of e-mail security protocols. Furthermore, we argue that this attack is entirely feasible in the networked environment in which these e-mail security protocols are used.

[*]Department of Computer Science, Columbia University; `jkatz@cs.columbia.edu`.
[†]Counterpane Internet Security, Inc.; `schneier@counterpane.com`.

2 Background

2.1 E-mail Encryption Protocols

The attack outlined herein is applicable to many different e-mail encryption protocols [17, 18]. In this paper, we explicitly consider attacks on OpenPGP [5, 7, 19] (the attack is also applicable to previous versions of PGP), S/MIME [16] (building upon CMS [10] and/or PKCS#7 [11]), PEM [14], and MOSS [6]. We refer the reader to the listed references for an in-depth description of these protocols; in this paper, we merely provide a high-level description necessary for a proper understanding of the attack. Specifically, the attack exploits the symmetric-key modes of encryption used, and therefore only this detail of the encryption protocol is presented.

Consider an e-mail message (or file) $M = M_1, M_2, \ldots$, where the message M is broken into a sequence of blocks of appropriate length for the underlying block cipher used. Encryption via the e-mail encryption protocols we consider operates (at a high level of description) as follows:

1. A random "session-key" K is generated.
2. M is encrypted using a symmetric-key encryption algorithm and key K, using some mode of encryption (we discuss below the details of the mode used). This gives ciphertext $c_0, c_1, c_2, \ldots$ (note the generation of the additional ciphertext block c_0).
3. The session-key K is encrypted using the recipient's public key. This is represented by $\mathcal{E}_{\text{pk}}(K)$.
4. The following message is sent to the recipient: $< \mathcal{E}_{\text{pk}}(K), c_0, c_1, \ldots >$.

The recipient, reversing the above steps, uses his private key to compute K; given K, the recipient can then use symmetric-key decryption to determine the original message M.

2.2 Chosen Ciphertext Attack

The attack presented here is known in the cryptographic literature as an adaptive chosen-ciphertext attack. The reader is referred elsewhere for formal definitions [1, 12], but a simple description is provided here. Assume an adversary intercepts ciphertext C and is trying to determine the underlying plaintext $P = \mathcal{D}(C)$ (where $\mathcal{D}(\cdot)$ refers to decryption of the ciphertext). Under an adaptive chosen-ciphertext attack, the adversary may submit ciphertexts $C_1, C_2, \ldots$ of his choice to a *decryption oracle* which then returns the corresponding plaintexts $P_1 = \mathcal{D}(C_1), P_2 = \mathcal{D}(C_2), \ldots$. The adversary is allowed to use the information thus obtained to determine the desired plaintext P. (Note that for the attack to be non-trivial the adversary is not allowed to submit the original ciphertext C to the decryption oracle.)

At first glance, this type of attack seems purely academic (when does an adversary have access to free decryption?), but consideration of the attack is of practical significance [8, 4, 2, 9]. One can readily think of examples in which information about the decrypted plaintext is available to the attacker. For instance, an adversary might be interacting with a computer which, when given some ciphertext, performs a specified action if the ciphertext is *valid* (i.e., whose decryption corresponds to some arbitrary plaintext) and does not perform the action otherwise, thereby allowing an adversary to distinguish valid ciphertexts from invalid ones. Such an attack on the RSA Encryption Standard PKCS#1 has been demonstrated [2], leading to a feasible attack on certain implementations of the SSL V.3.0 protocol. A similar attack, called a "reaction attack," has been used to break several coding-theory based public-key cryptosystems [9]. In yet another example [8], the adversary communicates with a party on the network who responds to ciphertext messages only if the decryption of the message corresponds to valid English text. This, too, gives the adversary information about the decrypted plaintext which may prove useful in cryptanalysis.

3 Details of the Attack

We stress that we do not expose any weaknesses in the public-key (typically RSA or ElGamal) or symmetric-key (IDEA, CAST, or 3-DES) algorithms used in the various encryption protocols. In fact, the attack is independent of the encryption algorithms used, and we therefore omit mention of specific algorithms when describing the attack. Rather, our attack focuses on the chaining mode used when encrypting messages longer than one block.

We begin with a description of the attack on the chaining mode used by OpenPGP [5]. OpenPGP uses a slight variation of Cipher Feedback (CFB) mode for symmetric encryption. Before encryption, the message M' is prepended by a 10-octet string. The first 8 octets are random, and the 9th and 10th octets are copies of the 7th and 8th octets, respectively. The resulting text $M = M_1, M_2, \ldots, M_k$ is parsed as a sequence of k blocks, each 64 bits long (for convenience, we assume a block cipher with 64-bit block size; the algorithm is similar for block ciphers with other block sizes). The OpenPGP variant of CFB-64 chaining mode is as follows ($\mathcal{E}_K(\cdot)$ represents application of the block cipher using session-key K):

```
**Encryption:** $c_0 = 0^{64}$
                for $i = 1$ to $k$:
                    $c_i = M_i \oplus \mathcal{E}_K(c_{i-1})$
                **Output:** $c_0, c_1, \ldots, c_k$
**Decryption:** for $i = 1$ to $k$:
                    $M_i = c_i \oplus \mathcal{E}_K(c_{i-1})$
                if ($c_0 = 0^{64}$) and (9th and 10th octets match 7th and 8th octets)
                    **Output:** $M_1, M_2, \ldots, M_k$
                otherwise
                    **Error**
```

To obtain the value of block M_i ($i > 2$), given ciphertext $< \mathcal{E}_{\text{pk}}(K), 0^{64}, c_1, \ldots, c_k >$, simply do the following:

1. Choose a (random) 64-bit number r.
2. Submit the ciphertext $< \mathcal{E}_{\text{pk}}(K), 0^{64}, c_1, c_2, c_{i-1}, r >$.
3. Receive back the decryption $M' = M'_1, \ldots, M'_4$, where $M'_4 = r \oplus \mathcal{E}_K(c_{i-1})$.
4. Compute $M_i = M'_4 \oplus r \oplus c_i$.

(Note that this also allows determination of block M_2.) If there is concern that the decrypted message will appear too similar to the original message, a random string can be inserted between c_2 and c_{i-1}; also, note that the last 6 octets of c_2 can be randomly chosen.

Other chosen ciphertext attacks are also possible. For example, submitting:

$$< \mathcal{E}_{\text{pk}}(K), 0^{64}, c_1, c_2^{16} r_1, c_3, r_2, \ldots, c_k, r_{k-1} >$$

(where c_2^{16} represents the first 16 bits of c_2, r_1 is a random 48-bit string, and $r_2, \ldots, r_{k-1}$ are random 64-bit strings) allows the adversary to compute the entire contents of the original message. The feasibility of any particular attack depends on the specifics of the "decryption oracle access" assumed available (which depends upon the behavior of the recipient of the original message; see below).

This type of attack is not limited to protocols using CFB mode. CBC mode, used by PEM [14] and CMS [10] is also vulnerable to a chosen ciphertext attack, as are all other "popular" modes of encryption [18] (including ECB, OFB, PCBC, and counter mode). Note further that CBC and CFB modes are "local" modes of encryption, in the sense that the i^{th} through the j^{th} blocks of the plaintext are completely determined by the $(i-1)^{\text{th}}$ through j^{th} ciphertext blocks. In addition, due to the reliance of these modes on XOR operations, individual bits of the i^{th} block (for CBC) or j^{th} block (for CFB) can be selected at will by an adversary mounting a chosen ciphertext attack. This allows the attack to work even though various redundancies are defined in these protocols.

4 Feasibility of the Attack

A chosen ciphertext attack on any e-mail encryption program is certainly feasible. Imagine a situation in which user U has configured his e-mail handler to automatically decrypt any incoming e-mails encrypted using PGP. An adversary A intercepts a PGP-encrypted message C sent to U, and wants to determine the contents of this message. Adversary A constructs C' according to above algorithm, and sends this message to U. Then, U's e-mail handler automatically decrypts C', and U reads the corresponding message M'. To U, the message appears garbled; U therefore replies to A with, for example, "What were you trying to send me?", but also *quotes the "garbled" message* M'. A receives the value M' which he wanted, and can use this to determine the original message. If U does not send A the garbled message; A can request it (for example, "I don't know what happened. Send me the file you decrypted and I'll try to figure it out.") This is not an unreasonable feat of social engineering.

Note that in this setting it is important that M' not look too similar to M, because otherwise U may become suspicious. For example, we may safely assume that the adversary will gain nothing if the submitted ciphertext decrypts to the same text as the original message (which is technically allowed under a chosen ciphertext attack), since in this case, the original recipient will certainly realize that something strange is going on.

In practice, messages are compressed before being encrypted. This makes the attack more difficult for the adversary (due to the redundancy a valid compressed message must contain), but by no means precludes an attack. Recall that the modes of operation used for encryption are "local" (as described above) and therefore the redundancy of the plaintext can be maintained via suitable alteration of the ciphertext. Furthermore, since the compression function is reversible and publicly known, the chosen ciphertext attack given above can be modified for this setting without difficulty.

5 Recommendations

We can immediately suggest some possible ways to prevent the attacks outlined herein. The simplest solution is for users not to reply to "garbage" e-mails by quoting the garbage message in their reply. This seems quite limiting—what if the message legitimately got corrupted in transit? It also relies too heavily on the behavior of users of the system.

Another solution is to demand that all encrypted messages be signed, and to not respond to unsigned messages with quotes from those messages. This is also limiting—it is common for people to send encrypted-but-unsigned PGP e-mail—and it is often good security practice not to attach a digital signature to an encrypted message. It also shares many of the problems of the previous solution.

Yet another solution is to have the e-mail decryption software store all session keys which have been used so far in messages sent to the user. (Actually, the e-mail encryption program should store a one-way hash of each session key, to avoid the problem of someone breaking into the user's files and obtaining a list of previously-used session keys.) Note that the chosen ciphertext attack described in Section 3 sends a previously-used session key as part of the ciphertext (in fact, it uses the same public-key encryption of the session key). The probability of this happening by chance is extremely low. A warning could be generated whenever a session key is repeated, to alert the user to be careful when responding to that particular message. Again, this solution is not completely satisfying. If encrypted e-mail becomes ubiquitous, storage of all the session keys used becomes cumbersome.

A possible future solution, and one which is more theoretically satisfying, is to have PGP use a mode of encryption which is itself secure under adaptive chosen ciphertext attacks [15, 3, 13]. Unfortunately, the various modes currently known to be secure in this setting each have their own drawbacks: expansion of the ciphertext length [13], requirement of additional primitives [15], and inefficiency [15, 3]. We hope that this paper will encourage additional research toward *efficient* modes of encryption secure under ciphertext attacks.

We recommend the following simple (though not provably secure) solution: append a hash of the message to the plaintext before encrypting. When decrypting, the software should check that the hash of the decrypted result matches the decrypted hash which is appended to the message. Encryption can be implemented in a number of ways: the one offering the most satisfying assurance is to encrypt the hash value separately from the message. For typical security requirements, we suggest using a hash with 128-bit output which can then be encrypted as a 128-bit block (using a 128-bit block cipher such as AES) or as two 64-bit blocks using DES. Another option for encrypting is appending the hash value to the message itself and encrypting the entire resulting string using a mode of operation as before. This offers security if the hash function is treated as a random oracle.

Note that this solution does not necessarily prevent a chosen ciphertext attack meeting the full technical definition in which the adversary may generate a second ciphertext which decrypts to the same plaintext as the original message (in which case the hash offers no additional protection). But, we can assume (as above) that the recipient will become suspicious upon receiving a message which decrypts to the same text as a previously-received message (especially if the contents of that message were confidential), making the "social engineering" aspect of this attack more difficult.

References

[1] M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway, "Relations Among Notions of Security for Public-Key Encryption Schemes," *Advances in Cryptology—CRYPTO '98 Proceedings*, Springer-Verlag, 1998, pp. 26–45.

[2] D. Bleichenbacher, "Chosen Ciphertext Attacks Against Protocols Based on RSA Encryption Standard PKCS#1," *Advances in Cryptology—CRYPTO '98 Proceedings*, Springer-Verlag, 1998, pp. 1–12.

[3] D. Bleichenbacher and A. Desai, "A Construction of a Super-Pseudorandom Cipher," Manuscript, February 1999.

[4] M. Blum, P. Feldman, and S. Micali, "Proving Security Against Chosen Ciphertext Attacks," *Advances in Cryptology—CRYPTO '88 Proceedings*, Springer-Verlag, 1990, pp. 256–268.

[5] J. Callas, L. Donnerhacke, M. Finney, and R. Thayer, "OpenPGP Message Format," RFC 2440, Nov 1998.

[6] S. Crocker, N. Freed, J. Galvin, and S. Murphy, "MIME Object Security Services," RFC 1848, Oct 1995.

[7] S. Garfinkel, *PGP: Pretty Good Privacy*, O'Reilly & Associates, 1995.

[8] S. Goldwasser, S. Micali, and P. Tong, "Why and How to Establish a Private Code on a Public Network," *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science* 1982, pp. 134–144.

[9] C. Hall, I. Goldberg, and B. Schneier, "Reaction Attacks Against Several Public-Key Cryptosystems," *Proceedings of Information and Communication Security*, Springer-Verlag, 1999, pp. 2–12.

[10] R. Housley, "Cryptographic Message Syntax," RFC 2630, Jun 1999.

[11] B. Kaliski, "PKCS #7: Cryptographic Message Syntax, Version 1.5," RFC 2315, Mar 1998.

[12] J. Katz and M. Yung, "Complete Characterization of Security Notions for Probabilistic Private-Key Encryption," *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing* 2000, to appear.

[13] J. Katz and M. Yung, "Unforgeable Encryption and Chosen-Ciphertext-Secure Modes of Operation," Manuscript, December 1999.

[14] J. Linn, "Privacy Enhancement for Internet Electronic Mail: Part I: Message Encryption and Authentication Procedures," RFC 1421, Feb 1993.

[15] M. Naor and O. Reingold, "On the Construction of Pseudo-Random Permutations: Luby-Rackoff Revisited," *Journal of Cryptology* 12(1): 29-66 (1999).

[16] B. Ramsdell, "S/MIME Version 3 Message Specification," RFC 2633, Jun 1999.

[17] B. Schneier, *E-Mail Security*, John Wiley & Sons, 1995.

[18] B. Schneier, *Applied Cryptography, 2nd Edition*, John Wiley & Sons, 1996.

[19] P. Zimmerman, *The Official PGP User's Guide*, MIT Press, 1995.