Programming Techniques and Data Structures

Ellis Horowitz Editor

Digital Signatures with RSA and Other Public-Key Cryptosystems

DOROTHY E. DENNING

ABSTRACT: Public-key signature systems can be vulnerable to attack if the protocols for signing messages allow a cryptanalyst to obtain signatures on arbitrary messages of the cryptanalyst's choice. This vulnerability is shown to arise from the homomorphic structure of public-key systems. A method of foiling the attack is described.

1. INTRODUCTION

George Davida [1] uncovered a potentially serious weakness in the basic protocol for signing messages using the RSA public-key cryptosystem [9]. Assuming that a cryptanalyst can get a user to sign arbitrary messages that may be meaningless, the cryptanalyst can decrypt ciphertext encrypted under the victim's public key or forge the victim's signature on a meaningful message. This is done by getting the victim to sign new messages derived from the intercepted ciphertext or chosen message. Although Davida refers to the attack as a "chosen signature" attack, it is actually a "chosen message" attack since the cryptanalyst chooses messages to be signed rather than signatures to be validated. The attack also works with other public-key systems.

The attack does not break the RSA system in the traditional sense whereby a cryptanalyst can obtain secret keys. Indeed, the attack is carried out without knowledge of the victim's private key. In this sense, the attack is much weaker than a "chosen plaintext" attack on a conventional cryptosystem which, if successful, breaks the system.

We first describe the attack with the RSA cryptosystem and show how it generalizes to other public-key systems. We then describe a signature-hashing technique, developed by Davies and Price [2], that appears to foil the attack for any public-key system.

Research supported in part by NSF Grant MCS80-15484.

2. THE BASIC RSA PROTOCOL

Let n be the modulus for the victim's RSA cryptosystem where $n = pq$ for large secret primes p and q (say 100 digits each, making n 200 digits); and let e and d be the public and private exponents respectively, where e and d are multiplicative inverses mod $\phi(n) = (p-1)(q-1)$. The public exponent e is used to encipher and validate signatures; the private exponent d to decipher and sign messages. To send the user a secret message M, the sender enciphers M by computing $C = M^e \bmod n$; the user deciphers the ciphertext C by computing $C^d \bmod n = M$. Similarly, the user signs a message M by computing $S = M^d \bmod n$; the receiver validates the signature S by computing $S^e \bmod n = M$. The security of the system rests on the assumption that a cryptanalyst cannot determine the factors p and q of n. [See [4] for a tutorial on the number theory behind the RSA system.]

3. THE POTENTIAL WEAKNESS

Suppose that a cryptanalyst has intercepted ciphertext C sent to the victim where $C = M^e \bmod n$. Davida [1] has shown how the cryptanalyst may be able to determine M without knowing the deciphering exponent d.

We first describe his method and then describe a simpler method found by Judy Moore.

ALGORITHM 1.
Davida's method for obtaining $C^d \bmod n = M$.

1. Factor C into $t > 1$ components, obtaining $C = C_1C_2, \cdots C_t$ (the components C_i need not be prime or prime powers—any decomposition of C will do). This implies that M also factors into t corresponding components $M_1, \ldots, M_t$ where
$$C = C_1C_2 \cdots C_t$$
$$= (M_1M_2 \cdots M_t)^e \bmod n$$
$$= (M_1)^e(M_2)^e \cdots (M_t)^e \bmod n,$$
and $M_i = C_i^d \bmod n$ $(i = 1, \ldots, t)$.
2. Get the victim to sign a message X and messages $XC_1 \bmod n, \ldots, XC_t \bmod n$. X can be a new message or a message previously signed by the victim. The messages $XC_1, \ldots, XC_t$ might be lines in some file the cryptanalyst requests the victim to sign line by line to acknowledge receipt. The signatures obtained are thus:
$$S = X^d \bmod n$$
$$S_i = (XC_i)^d \bmod n \ (i = 1, \ldots, t).$$
3. Compute the multiplicative inverse $S^{-1} \bmod n$ of the signature S, getting
$$S^{-1} = X^{-d} \bmod n.$$
4. Multiply this by each of the S_i to obtain the M_i:
$$S^{-1}S_i \bmod n = X^{-d}(XC_i)^d \bmod n = C_i^d \bmod n = M_i.$$
5. Compute the product $M_1M_2 \cdots M_t \bmod n = M$.

The attack can also be made using $X = 1$. Then the cryptanalyst needs only the t signatures $S_i = C_i^d \bmod n = M_i$. This attack, however, may be easier to detect since the factors of C and M are exposed.

Obviously the attack would not be feasible if a complete factorization of C were required in Step 1, since we are assuming that factoring large (e.g., 200 digit) numbers is infeasible (if not, then RSA could be broken). Even so, factoring is costly, so we are more interested in Moore's method which does not require any decomposition.

The multiplicative inverse of S computed in Step 3 can be computed efficiently in $O(\log n)$ time using an extended version of Euclid's algorithm (e.g., see [4]). In the unlikely event that S is not relatively prime to n, S does not have a unique inverse mod n. But in this case, the cryptanalyst can easily factor n because S will be a multiple of p or q, whence $gcd(S, n) = p$ or q.

In Step 2, the cryptanalyst can pick an arbitrary message S and compute $X = S^e \bmod n$ rather than asking the victim to sign X, since this implies that $S = X^d \bmod n$. This reduces the amount of cooperation required from the victim and is used in Step 1 of Algorithm 2, the simpler approach found by Moore. Because the victim does not know X or S, it is unnecessary to decompose C to conceal the attack, that is, $t = 1$ can be used. Thus, Moore's attack requires only one signature from the victim:

ALGORITHM 2.
Moore's method for obtaining $C^d \bmod n = M$.

1. Pick an arbitrary S and compute $X = S^e \bmod n$; this implies
$$S = X^d \bmod n.$$
2. Get the victim to sign the message $XC \bmod n$, obtaining the signature
$$S_1 = (XC)^d \bmod n.$$
3. Compute $S^{-1} \bmod n = X^{-d} \bmod n$.
4. Multiply S^{-1} by S_1 to obtain M:
$$S^{-1}S_1 \bmod n = X^{-d}(XC)^d \bmod n = C^d \bmod n = M.$$

Under the assumption that the RSA system in cryptographically strong (computationally infeasible to break), Moore's method is optimal in the sense of requiring the minimal number of signatures from the victim. If it were not, then we could decrypt ciphertext without any cooperation from the victim, thereby breaking RSA.

Because both algorithms compute $C^d \bmod n$, they may also be used to forge the victim's signature on a message C chosen by the cryptanalyst. If the cryptosystem is used for signatures only and not for message secrecy, or if separate keys (values of e, d, and n) are used for secrecy and for signatures, the attack can still be used to forge signatures, though it cannot be used to decrypt ciphertext.

4. GENERALIZING THE RESULTS TO OTHER PUBLIC-KEY SYSTEMS

Consider an arbitrary public-key cryptosystem with private deciphering (signature) transformation D and public enciphering (signature validation) transformation $E = D^{-1}$. Moore's method extends to the system if the message space forms a group with binary operator o and identity element 1; the signature space forms a group with a binary operator, also denoted by o, and identity 1; and the deciphering transformation D is a homomorphism from the message group to the signature group, that is, the following properties hold for all messages X, Y, and Z:

1. $X \circ (Y \circ Z) = (X \circ Y) \circ Z$	(Associativity—for Step 4)
2. $X \circ 1 = 1 \circ X = X$	(Identity—for Step 4)
3. $X \circ X^{-1} = X^{-1} \circ X = 1$	(Inverses—for Steps 3 and 4)
4. $D(X \circ Y) = D(X) \circ D(Y)$	(Homomorphism—for Step 4)

Algorithm 3 computes $D(C)$ to decrypt C or forge the victim's signature on C:

ALGORITHM 3.
Generalization of Moore's method to obtain $D(C)$.

1. Pick S and compute $X = E(S)$; this implies $S = D(X)$.
2. Get the victim to sign the messae $X \circ C$. The signature is $S_1 = D(X \circ C)$.
3. Compute S^{-1}.
4. Compute

$$\begin{aligned} S_2 &= S^{-1} \circ S_1 \\ &= S^{-1} \circ D(X \circ C) = S^{-1} \circ (D(X) \circ D(C)) \\ &= S^{-1} \circ (S \circ D(C)) = (S^{-1} \circ S) \circ D(C) \\ &= 1 \circ D(C) = D(C). \end{aligned}$$

The RSA system fits this general pattern, where both the message and signature groups are defined by the integers relatively prime to n together with multiplication. The deciphering transformation is a homomorphism because

$$(XY)^d \bmod n = [(X^d \bmod n)(Y^d \bmod n)] \bmod n.$$

(Similarly, the enciphering transformation is a homomorphism.)

It is not surprising that a cryptosystem for which the deciphering transformation is a homomorphism is vulnerable to certain types of attack. Rivest et al. [10] showed that such cryptosystems can have inherent weaknesses.

In a signature-only system, messages can be signed but not encrypted for secrecy, since computing $D(E(X))$ will not usually restore X. For such systems, the attack is therefore useful only for forging signatures. Note that in this case, the transformation D need not be a homomorphism as long as E is a homomorphism. This is because the signature S_2 computed in Step 4 of Algorithm 3 will be accepted as a valid signature, though it can differ from that which the victim would compute, that is,

$$\begin{aligned} E(S_2) &= E(S^{-1} \circ S_1) = E(S^{-1}) \circ E(S_1) \\ &= E(S)^{-1} \circ E(S_1) = X^{-1} \circ (X \circ C) = C. \end{aligned}$$

Shamir's signature-only knapsack scheme [11] (see also [4]) fits this pattern, as shown by DeMillo and Merritt [3]. Here, the signature validation transformation E is given by $E(S) = SA \bmod n$, where n is a k-bit prime, A and S are vectors of length $2k$ whose elements are k-bit integers, and SA denotes the scalar product. The signature group is defined by the integer vectors of length $2k$ with vector addition and the 0 vector as identity. The messages C and X are integers mod n, and the message group is defined by the integers mod n with addition and identity 0. Although D is not a homomorphism, E is a homomorphism from the signature group to the message group, since for all signatures S_1 and S_2:

$$(S_1 + S_2)A \bmod n = (S_1A \bmod n + S_2A \bmod n) \bmod n.$$

Algorithm 4 shows how a signature can be forged in Shamir's system.

ALGORITHM 4.
DeMillo's and Merritt's Method for Forging a Signature S_2 on C with Shamir's Signature System.

1. Pick X and get the victim to return the signature S such that $SA \bmod n = X$.
2. Get the victim to return a signature S_1 on $X + C$ such that $S_1A \bmod n = X + C$.
3. Compute $S^{-1} = -S$.
4. Compute the signature $S_2 = -S + S_1$. S_2 is a valid signature of C because

$$\begin{aligned} E(S_2) &= E(-S + S_1) = (-S + S_1)A \bmod n \\ &= (-SA \bmod n + S_1A \bmod n) \bmod n \\ &= [-X + (X + C)] \bmod n = C. \end{aligned}$$

DeMillo and Merritt also consider similar attacks on variants of the RSA system. These systems all have an underlying homomorphic structure (though not explicitly identified as such in their paper), which explains their vulnerability to this general method of attack.

5. AN IMPROVED SIGNATURE SCHEME

For the attacks to succeed, the cryptanalyst must be able to get the victim to sign essentially arbitary messages that are supplied by the cryptanalyst and are not likely to be meaningful. To protect against such attacks, users can sign only meaningful messages, but this places the burden of security entirely on the user.

A better strategy is to strengthen the signature system to make the attacks impossible. We now describe a method of doing this that transforms all messages with a one-way public function h before they are signed. A message M is thus signed by computing $S = D(h(M))$.

The function h should satisfy four properties:

1. h should destroy all homomorphic structure in the underlying public-key cryptosystem; that is, $h(X \circ Y) \neq h(X) \circ h(Y)$ must hold. Moreover, for almost all X and Y, $D(h(X \circ Y)) \neq D(h(X)) \circ D(h(Y))$ should hold. Then the cryptanalyst cannot factor out the X or Y in a signature $D(h(X \circ Y))$.
2. h should be computed over entire messages (rather than on a block-by-block basis). This will make it difficult for a cryptanalyst to obtain signatures by inserting blocks into a file that otherwise looks legitimate.
3. h should be one-way so that messages are not disclosed by their signatures. This is needed when both secrecy and authenticity are desired.
4. h should have the property that for any given message X and value $h(X)$, it is computationally infeasible to find another message Y such that $h(Y) = h(X)$. This is needed to prevent forgeries since the value $h(X)$ can be obtained from a signature $S = D(h(X))$ by applying the public function E to S.

Applying the transformation h to a message before signing has an additional benefit. Because the transformations $E(\ )$ and $D(h(\ ))$ are not inverses, a cryptanalyst cannot hope to decrypt an intercepted ciphertext message C by getting a signature on C.

The practice of transforming messages before signing was suggested by Davies and Price [2] who designed a protocol for signing secret messages using a one-way public hashing function h that conceals messages and prevents forgeries. Their function h blocks a message M into 56-bit blocks $M_1, \ldots, M_r$ and computes a digest (64 bits)

$$H = h(M, I) = E_{M_r} \circ \ldots \circ E_{M_1} \circ E_{M_r} \circ \ldots \circ E_{M_1}(I),$$

where E_{M_i} is the DES-enciphering algorithm keyed to block M_i, I is a random 64-bit initialization seed for the DES, and "o" here denotes function composition. Because the function h can be computed both forwards and backwards (by using the deciphering transformations D_{M_i}) for an arbitrary message M, the message must be repeated in the keys to prevent a "meet in the middle" forgery (compute forwards from I using 2^{32} variations of the first half of the desired message and backwards from H using 2^{32} variations of the second half of the message; sort the results to find a match, which is likely to occur since there are 2^{64} ways of pairing the values in the two sets).

Wolfgang Bitzer has suggested an improved hashing function that foils the meet-in-the-middle attack with a single pass over the message. His hashing function is given by

$$H = h(M, I) = Z_{r+1},$$

where

$$Z_{i+1} = E_{Z_i' \oplus M_i}(Z_i) \quad (i = 1, \ldots, r)$$

$$Z_1 = I,$$

Z_i' consists of 56 bits selected from Z_i, and "$\oplus$" denotes "exclusive or." The meet-in-the-middle attack is prevented because it is not possible to compute backwards through the function, i.e., compute Z_i from Z_{i+1}.

Both DES-based hashing functions would destroy the multiplicative structure of the RSA system and the additive structure of knapsack systems. They almost certainly would destroy the homomorphic structure of any underlying cryptosystem.

Using the hashing function, the message M is signed as $S = D(H \| I)$, where "$\|$" denotes concatenation, and the 128-bit block $(H \| I)$ is replicated as many times as necessary to fill the input block for the signature transformation D. A signature S on an alleged message M is validated by first computing $E(S) = D^{-1}(S) = (H \| I)$; next computing $h(M, I)$ using the public function h and the alleged message M; and finally comparing $h(M, I)$ with H.

The message M can be transmitted either as cleartext (if secrecy is not needed), as ciphertext encrypted using the receiver's public key, or as ciphertext encrypted using a secret key shared by the sender and receiver (if a conventional cryptosystem is used for message secrecy, with the public-key system reserved for signatures and key exchange).

The hashing function has two important advantages besides protecting against signature attacks:

1. It separates the signature transformation from the secrecy transformation, allowing secrecy to be implemented with a one-key system or to be skipped [2]. Yet the separation is achieved without much message expansion, since each signature is a single block.
2. It conceals messages so that signatures can be publicly disclosed without revealing their corresponding messages. This is important for recovering from compromises or direct disclosure of private keys. Let D_A be the signature key of user A. In order that a signature S of A can remain valid after D_A is compromised or deliberately disclosed, S should be bound to A's current public key E_A, time-stamped, and signed by the public key server (notary public) [8] giving a "signature certificate" [5] $G = D_P(T, A, E_A, S)$, where D_P is the signature key of the public key server, and T is the time stamp. And in order that S can remain valid even if D_P is compromised, G should be kept in a public log. For this reason, it is important that S conceal the message signed and have minimal storage requirements. This is achieved with the hashed signature method. (For further discussion of this, see [5].)
3. It can provide a more efficient method of signing messages, since the public key transformation is applied to just one block of data. The RSA transformation, for example, is several orders of magnitude slower per bit than the DES, so using the DES to hash a long message down to a single block is considerably faster than applying RSA to the entire message.

6. CONCLUSIONS

Davida's discovery demonstrates the fundamental importance of encryption protocols. It is not enough to have an encryption algorithm that is computationally hard to break; the procedures for using the algorithm must also withstand attack. We have identified several properties that should be satisfied by any signature system; in particular, it should destroy any homomorphic structure in the underlying public-key algorithm. The signature scheme described here appears to satisfy these properties. Further research along the lines initiated by Dolev and Yao [7] and DeMillo et al. [3] may lead to techniques for proving the security of signature schemes.

The attacks on public-key signature systems are instances of a general method of attack, which we call the "many-time pad" [6]. Many-time pad attacks pad the input to a query to conceal a request for sensitive data, and then remove the effect of the padding from the output by exploiting an underlying homomorphic structure in the problem domain. "Tracker" attacks

(e.g., see [4]) on statistical databases are also instances of the many-time pad.

Acknowledgments This note evolved from discussions with George Davida, who first brought the signature attack on RSA to my attention; Judy Moore, who told me of her simpler attack; and Michael Merritt, who independently with Richard DeMillo identified a mathematical structure in the attacks similar to the one described here. Many others have provided helpful suggestions, including Bob Blakley, Peter Denning, Ron Graham, Ron Rivest, and the students in my cryptography class. Donald Davies brought Bitzer's hashing function to my attention. Finally, the referees provided many constructive comments.

REFERENCES

1. Davida, G.I. Chosen signature cryptanalysis of the RSA (MIT) public key cryptosystem. Tech. Rept. TR-CS-82-2, Dept. of Electrical Engineering and Computer Science, Univ. of Wisconsin, Milwaukee, WI., Oct. 1982.
2. Davies, D.W. and Price, W.L. The application of digital signatures based on public key cryptosystems. Tech. Rept. NPL Report DNACS 39/80, National Physical Lab., Teddington, Middlesex, England, Dec. 1980.
3. DeMillo, R., Lynch, N.A. and Merritt, M.J. Cryptographic protocols. Proc. SIGACT Conf., 1982.
4. Denning, D.E. *Cryptography and Data Security.* Addison-Wesley, Reading, MA., 1982.
5. Denning, D.E. Protecting public keys and signature keys. *IEEE Computer 16,* 2 (Feb. 1983), 27–35.
6. Denning, D.E. The many-time pad: Theme and variations. *Proc. 1983 Symp. Security and Privacy,* IEEE Computer Society, Apr. 1983.
7. Dolev, D. and Yao, A.C. On the security of public key protocols. Proc. 22d *Ann. Symp. Foundations of Comput Sci,* IEEE Computer Society, 1981.
8. Merkle, R.C. Protocols for public key cryptosystems. *Proc. 1980 Symp. Security and Privacy,* IEEE Computer Society, April, 1980, pp. 122–133.
9. Rivest, R.L., Shamir, A. and Adleman, L. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM 21,* 2 (Feb. 1978), 120–126.
10. Rivest, R.L., Adleman, L. and Dertouzos, M.L. On data banks and privacy homomorphisms. In *Foundations of Secure Computation,* R.A. DeMillo et al., Eds., Academic Press, New York, 1978, 169–179.
11. Shamir, A. A fast signature scheme. Tech. Rept. MIT/LCS/TM-107, MIT Lab. for Computer Science, Cambridge, Mass., July 1978.

CR Categories and Subject Descriptors: E.3 [**Data**]: Data Encryption—public-key cryptosystems
General Term: Security
Additional Key Words and Phrases: cryptanalysis, cryptographic protocol, hashing, homomorphism

Received 11/82; revised 7/83; accepted 11/83

Author's present address: Dorothy E. Denning, SRI International, Computer Science Lab, 333 Ravenswood Ave., Menlo Park, CA 94025