Appendix D

Message Authentication, Public-Key Ciphers, and Digital Signatures

Message Authentication

An "authentic" message is one that has arrived exactly as it was sent (without errors or alterations), is not a replay of a previous message, and comes from the stated source (not forged or falsified by an imposter or fraudulent altered by the recipient). [1] Encipherment in itself does not automatically authenticate a message: it protects against passive eavesdropping automatically, but does not protect against some forms of active attack. [2]

Encipherment algorithms can be used to authenticate messages, however, and the algorithm used in the Data Encryption Standard (DES) is the most widely used cryptographic basis for message authentication. As discussed in more detail later, there are profound differences in using a symmetric cipher, such as the current DES algorithm, rather than an asymmetric one like the RSA algorithm named after its inventors: Ronald Rivest, Adi Shamir, and Leonard Adelman. Use of a symmetric cipher for message authentication can only protect against third parties and not against fraud by either the sender or receiver (both of whom know the secret key), while an asymmetric algorithm can be used to resolve disputes between the sender-receiver pair.

As the uses of electronic media for financial and business transactions have proliferated, message authentication techniques have become increasingly important and have evolved from simple pencil-and-paper calculations to sophisticated, high-speed hardware processors.

In general, the various message authentication schemes that are used can be grouped together according to whether they are based on public knowledge or, at least in part, on secret knowledge. Among the former are message authentication using check fields, [3] parity checks, [4] and cyclic redundancy checks. [5] These share a common weakness: unauthorized or fraudulent modifications may go undetected because they are accompanied by matching, yet fraudulent, authentication parameters that can be calculated by unauthorized parties because the authentication parameters are not secret.

Using secret authentication parameters known only to the sender and receiver permits a stronger form of message authentication because the check field data cannot be forged by a third part y unless

[1] For a thorough discussion of message authentication and the various techniques used to authenticate messages, see Davies & Price, *Security for Computer Networks: An Introduction to Data Security in Teleprocessing and Electronic Fund Transfers* (New York, NY: J. Wiley & Sons, 1984) ch. 5. The descriptions of authentication techniques in this section follow Davies & Price closely.

[2] Davies & Price describe passive attack as eavesdropping and active attack as the falsification of data and transactions through such means as: 1) alteration, deletion, or addition; 2) changing the apparent origin of the message; 3) changing the actual destination of the message; 4) altering the sequence of blocks of data or items in the message; 5) replaying previously transmitted or stored data to create a new false message; or 6) falsifying an acknowledgment for a genuine message. (See Davies & Price, op. cit., pp. 119-120.)

[3] Check-field techniques are designed to ensure that stored or transmitted information has been prepared correctly. A check field is a simple function of the numerical characters in the important fields of the message that makes it highly likely that the most common types of mistakes and errors will be detected. The use of check fields does not safeguard against deliberate fraud by data-entry operators or others; because the check sum function and the data used to create it are not secret, a data-entry operator could calculate the "correct check sum and transmit it along with a fraudulently altered message. Also, it is possible to generate false messages that have the same calculated check sum value as the original message. (See Davies & Price, op. cit., pp. 121-122.)

[4] Parity checks can be used to detect accidental errors during transmission, usually either by using an eighth "parity bit with each seven-bit message word or by providing longitudinal parity checks using modulo-2 addition on successive words. Parity checks are weak in that multiple errors and/or missing blocks can sometimes go undetected. (See Davies & Price, op. cit., p. 122.)

[5] According to Davies & Price, cyclic redundancy checks are the best-known error detection method and offer strong protection against accidental error. However, the procedure for creating the check data via a predetermined polynomial is public knowledge. Therefore, the checks do not provide strong protection against an active attack. In this form of message authentication, the cyclic check operation calculates the check data by dividing the polynomial formed by a block of message bits by the predetermined check **polynomial** and using the "remainder" from the polynomial division as a check field. The check field is appended to the message block and transmitted with the message. Upon its receipt, the recipient performs the same polynomial division operation on the message and compares the remainder with the transmitted check field to authenticate the message. The cyclic check does not protect against active attack because the means of creating the check data—the polynomial division operation—is not secret. An active wiretapper can divert the message, alter it, calculate a new check field using the correct predetermined polynomial, and retransmit the altered message with the new check field appended. The message will appear to be authentic when the recipient compares the check fields. (See Davies & Price, op. cit., pp. 122-123.)

the secret parameters are compromised. A different secret parameter is usually required for each sender-receiver pair. The logistics for distributing this secret information to the correct parties is analogous to key distribution for encryption. Compromise of the secret parameters invalidates the integrity of the safeguarding function because it could then be forged. (See ch. 4.)

In the most general sense, an authentication function based on secret knowledge can be constructed from a public authenticator algorithm and a secret authentication key.[6] Examples of authenticators based on secret keys include the Decimal Shift and Add (DSA) algorithm,[7] and the proprietary S. W. I.F.T. (Society for Worldwide Interbank Financial Telecommunications) and Data Seal algorithms, which use binary arithmetic.[8] Although authentication can be based on encryption (the DES algorithm, for example, is widely used for message authentication), the strict requirements for an authenticator algorithm differ from those for an encryption algorithm because authentication does not require the existence of an inverse function (i.e., decryption). It is also possible to base message authentication on secret numbers used in conjunction with special one-way functions. [9]

Encryption alone is not always sufficient to completely authenticate a message. If decryption of an encrypted message yields "sensible" plaintext, without garbled portions, then there is reasonable certainty that the message was originated by the other "authorized" holder of the secret key. However, some types of message alterations can go undetected. [10] A more robust authentication method (than DES encryption alone, for example) is to use DES hardware in an appropriate mode of operation in order to create a message authentication code. [11]

When DES hardware is used for authentication, it is operated in either the cipher block chaining (CBC) or cipher feedback (CFB) mode; the chaining or feedback operation ensures that the authenticator, selected from the last state of the DES hardware output register, is a function of the entire message stream input to the DES device. [12] The authenticator is appended to the message and transmitted along with it. The recipient removes the authenticator from the received message and calculates his or her own value of the authenticator using the secret key and initialization vector shared with the sender. If the two authenticator values are the same, then there is increased assurance that the message is authentic.

The message can be transmitted in plaintext without compromising its authenticity. If the message is altered by a third party who does not use the secret DES key to calculate a forged authenticator to append to the altered message, then the authenticator calculated by the receiver will not have the same value as the one transmitted with the message. However, because both sender and receiver know the secret parameters, either could fraudulently alter the message and deny having done so. This type of dispute can be resolved through use of an asymmetric cipher, as will be discussed below in the sections on public-key ciphers and digital signatures.

If privacy as well as authentication is required, then one scheme for encrypting and authenticating a message involves sequential use of DES using two different secret keys: one to calculate the authenticator (called the message authentication code or MAC), and one to encrypt the message. These operations can be performed in either order; the ANSI X9.23 standard requires that the MAC be calculated before encryption. " Even the MAC

"For mathematical and functional descriptions of authenticator functions, see: Davies & Price, op. cit., pp. 123-135; and Et. R. Jueneman, S.M. Matyas, and C. H Meyer, " 'Message Authentication, " *IEEE Communications Magazine*, vol. 23, No. 9, September 1985,

[7] DSA is based on parallel computations performed by the sender and receiver; the starting point for the computations are two secret 10-digit decimal numbers. The message to be authenticated is treated as a string of decimal digits, thus DSA requires that alphabetic characters be encoded into numeric form, although the numeric content of a message (e. g., the value of a financial transaction) does not require any conversion. According to Davies & Price (see pp. 127-130 for an example of DSA), the algorithm can be implemented using a programmable decimal calculator.

"Because these are proprietary, they are not available for use as a published standard.

"A one-way function has the special property that. although the function itself is relatively easy to compute, its inverse is quite difficult to compute— i.e., even if the values of authenticators for many messages are known, it is almost impossible to recover the text of a message given only the value of its authenticator. Some, hut not alE, "hash functions '— functions that appear to generate random outputs from nonrandom inputs — are suitable for message authentication.

[10] See Davies & price, op cit., pp. 134-135 for examples. Jueneman, et al , also discuss strengths and weaknesses of various authentication and manipulation detect ion techniques.

[11] Strictly speaking, any block encryption algorithm could be used. However, in practice, the cipher used is DES because the algorithm is readily available in hardware form. The DES algorithm is relatively slow in software form, which makes the hardware form much more convenient for data transmission.

[12] In the CBC mode, the authenticator is the most significant *n* bits from the last block output by the device. In the CFB mode, the DES device is operated one additional time after the last message block is input, and the authenticator is selected as the most significant *n* bits of the final output block. The length of the authenticator (usually 32 bits for EFT authentication, according to the standard) is determined jointly by the sender and receiver.

[13] If the MAC checks the ciphertext, then an adversary is able to mount a known plaintext attack on the key used for authentication, If, however, the MAC checks the plaintext, then an adversary must break both the MAC kc)' and the encryption key in order to send fraudulent messages,

and encryption do not safeguard against replay of messages (e.g., electronic fund transfers). Therefore, various message sequence numbers or date and time stamps are usually incorporated into the text of the message. The ANSI X9.9 standard requires a message identifier field to prevent replay.

Public-Key Ciphers

A symmetric cypher is an encryption method using one key, known to both the sender and receiver of a message, that is used both to encrypt and to decrypt a message. Obviously, the strength of a symmetric cipher depends on both parties keeping the key secret from others. With DES cipher, for example, for which the algorithm is known, revealing the encryption key permits the message to be read by any third party.

An asymmetric cypher is an encryption scheme using a pair of keys, one to encrypt and a second to decrypt a message.[14] A special class of asymmetric ciphers are public-key ciphers, for which the encrypting key need not be kept secret to ensure private communication.[15] Rather, Party A can publicly announce his encrypting key, PK_A, allowing anyone who wishes to communicate privately with him to use it to encrypt a message. Party A's decrypting key, SK_A, is kept secret, so that only A (or someone else who has obtained the secret decrypting key) can easily convert messages encrypted with PK_A back into plaintext.[16]

Knowing the public encryption key–even when the encrypted message is also available-does not make computing the secret decrypting key easy, so that in practice only the authorized holder of the secret key can read the encrypted message.[17] However, with the encrypting key being publicly known, a properly encrypted message can come from any source, so there is no guarantee of its authenticity.

It is also crucial that the public encrypting key be authentic. An imposter could publish his own key, PK_I, and for example, pretend it came from A in order to read messages to A, which he could intercept and then read using his own SKI. Therefore, the strength of the public-key cipher rests on the authenticity of the public-key and the secrecy of the private key. A variant of a public-key system allows a sender to authenticate messages by "signing" them using an encrypting key, which (supposedly) is known only to him. This is a very strong means of authentication and is discussed further in the following section on digital signatures.

Davies and Price[18] review and illustrate the functional requirements for a general public-key cryptosystem. A brief overview follows here, but a detailed description of the underlying mathematics is beyond the scope of this appendix,

If encipherment is performed by some function E{ K,) and decipherment by another D(K_d), then in order to make the decipherment function the inverse of encipherment, the encrypting key, K_e, and the decrypting key, K_d, must be related somehow. Suppose both keys are derived from a randomly selected starting key, or seed key, K, such that K_e = F(K_s) and K_d = G(K_s), where the functions F, G, D, and E defined above are published. Party A would then select a K_s (which would be kept secret), use it to calculate K_e and K_d, and publish K_e, as his public key, PK_A, while keeping K_d secret as the secret key, SK_A.

If P is the plaintext message and C is the encrypted message, then C =E(P) and P= D(E(P)); that is, D(E(P)) must be the inverse of E. However, because E is really the function E(K,) and is public, the function E must not be readily invertible or else an opponent can readily calculate P given C and E. This property is described as making E

[14] See Davies & Price, op. cit., ch. 8, for a more complete discussion of asymmetric and public-key ciphers.

A discussion of the underlying principles of public-key ciphers, including examples of the RSA and knapsack algorithms, is given in: Martin E. Hellman, "The Mathematics of Public-Key Cryptography, " *Scientific American, vol.* 241, No. 2, August 1979, pp. 146-157.

A pictorial example of the RSA public-key method can be found in *Computer Security* (one of the *Understanding Computers* series) (Alexandria VA: Time-Life Books, 1986), pp. 112-117.

[15] The public-key concept was first proposed by Whitfield Diffie and Martin Hellman in "New Directions in Cryptography, " *IEEE Trans. Information Theory, IT-22, 6,* November 1976, pp. 644-654. Diffie and Hellman also described how such a public-key cryptosystem could be used to "sign" individual messages and to simplify the distribution of secret keys. Their work was the basis for Rivest, Shamir, and Adelman's practical implementation of such a system in 1978, called the RSA cipher. Some ciphers proposed for public-key systems have subsequently been broken. For example, the Diffie-Hellman "discrete exponentiaf" cipher was broken several years later by Donald Coppersmith of IBM [G. Kolata: "Another Promising Code Falls, " *Science,* vol. 226, Dec. 16, 1983, p. 1224]. The "trap-door knapsack" cipher, another public-key cipher proposed in 1976 by Hellman and Ralph Merkle, was broken by Shamir and Adelman in 1982. (See *Computer Security,* op. cit., pp. 100-101; and Hellman's 1979 article in *Scientific American.)*

[16] This *section* uses *the notation* PK for "public key" (usually, the encrypting key) and SK for "secret key" (usually, the decrypting key). For A and B to have two-way communication, two pairs of keys are required: the "public" encryption keys PK_A and PK_B, and the secret decryption keys SK_A and SK_B.

[17] Use of the two keys might also be used to separate access to "read" and ' 'write' data functions. For example, by controlling dissemination of the encryption key, one might control write access; by controlling dissemination of the decryption key, read access.

[18] Davies & Price, op. cit., ch. 8.

(and also the function F, which generates K, from K_s one-way functions that cannot readily be inverted.

The requirements that E be a noninvertible, one-way function and that D(E(P)) be its inverse are reconcilable when E is not invertible without knowledge of K_e, but the inverse of E is readily obtained using the secret key K_d. Thus, knowledge of the secret key is a' 'trapdoor, which makes the inverse of E simple to implement. E(P), where E = E(K_e), is a one-way function with a trapdoor D(E(P)), which allows it to be inverted. Knowledge of K_d springs the trapdoor. [19]

A public-key cryptosystem consists of encryption and decryption functions, together with methods for generating pairs of keys from the random seed values. The one-way property of a "one-way function, " such as E, is really only a matter of computational complexity. The encryption function should be relatively easy to carry out, given K_e. and E, but given the ciphertext C = E(P), the plaintext P =D(E(P)) should be very hard to calculate and should require a very large number of steps, unless K_d is known.

In principle, it should be possible to calculate values of C for many values of P and then to sort and tabulate the pairs of (Pi, Ci) to obtain an explicit inversion of E. Because this type of exhaustive search process requires a large number of computational steps and large computer memory size, both of which grow exponentially with the key size, E is effectively a one-way function if the explicit inversion requires a very large number of (P,C) pairs.

Like all of modern cryptography, public-key cryptosystems rely heavily on mathematics and, in particular, on number theory. The RSA cipher is based on modular arithmetic[20] and the trapdoor knapsack cipher is based on combinatorial mathematics as well, The mathematical problems underlying the RSA cipher and the knapsack public-key cipher belong to a class of problems called "nondeterministic, polynomial-time problems, " or NP problems. The computational burden of finding a solution to the hardest NP problems, using published methods, grows very rapidly as the size of the problem increases. It is strongly believed (but not proved) that the burden must grow very rapidly, no matter what method of solution is used. However, once the solution is found, it can be checked very easily. [21] Even so, it is possible that advances in mathematics and computer science may undermine public-key cryptosystems based on "one-way" functions. One instance of this was the "breaking" of the trapdoor knapsack cipher. Box G describes this cipher.

The knapsack cipher system was thought to be effectively unbreakable (computationally but not unconditionally secure) and Merkle issued an open challenge to cryptologists in 1976 to break it, In 1982, Adi Shamir at the Massachusetts Institute of Technology (MIT) broke the cipher analytically. Soon afterward, Leonard Adelman, a former colleague of Shamir, used Shamir's method and an Apple II computer to break an example of the knapsack cipher.

Another public-key system, called the RSA system, was announced in 1978. The RSA system is computationally more complex to implement than the trapdoor knapsack cipher and it has not yet been broken. Also, the RSA system does not expand the plaintext message the way the knapsack cipher does, Message expansion occurs with the knapsack cipher because the sum of the "hard" knapsack vector used in the knapsack public key is larger than the sum of the "easy" vector used in the secret key. Therefore, more binary bits are required to represent the ciphertext than to represent the plaintex!.

The RSA Public-Key System

The RSA public-key system is thought to be the most computationally secure, commercially available public-key system. It also enables the prob-

[19] See Davies & Price, op. cit., ch. 8 for a more thorough explanation. Diffie and Hellman introduced the concept of trapdoor one-way functions in their 1976 paper (op. cit.), but did not present any examples. In their 1978 paper, Rivest, Shamir, and Adelman presented their implementation of a public-key system using a one-way trapdoor function. See also R.L. Rivest, A. Shamir, and L. Adelman "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," *Communications of the ACM*, vol. 21, No. 2, February 1978; and Hellman's article in the August 1979 issue of *Scientific American* op. cit.

[20] Finite arithmetic with modulus *m* (modular arithmetic) has the operations of addition, multiplication, subtraction, and division defined. A *prime number*—e.g., *3, 5,* or 7 in modulo 10 arithmetic—has no factors other than 1 and itself. Finite arithmetic with a *prime modulus p* has the additional property that multiplication always has an inverse. This property is crucial for cryptography}'.

I n modulus 10 arithmetic, for example, the number 57 would be represented by its remainder, 7 : 57 10 = 5, with a remainder of 7. More general}', the remainder always has a value between 0 and (m 1), where m is the modulus Thus, in modulus 3 arithmetic, 57 would be represented by the remainder of 0; in modulus l 1 arithmetic by the remainder of 2, etc

The exponential function a^x in modulus *p* arithmetic is valuable as a one-way function: calculating the exponential $y = a^x$ is easy, but calculating its inverse, $x = \log_a y$ is difficult for large *p*.

[21] See Hellman's article in the August 1979 *Scientific American* op. cit.

Box G.—The Trapdoor Knapsack Cipher

The trapdoor knapsack system was proposed by Ralph Merkle and Martin Hellman in 1976.[1] The "knapsack puzzle" or "knapsack problem" is well-known in mathematics and can be summarized briefly as the following problem:

> Suppose you are given a set of N weights of assorted (and known) integer sizes. You want to use them to balance a knapsack that holds an unknown combination of the same weights. You are given the values of the set of integer weights (called the knapsack vector) and the weight to be matched (called the knapsack total). Find the subset of the N weights that will exactly balance the knapsack!
>
> Although it is possible to find examples of this problem that are fairly easy to solve by exhaustive search— when N is a small number or when each weight is heavier than the sum of the proceeding weights, for example–all of the known methods of solving the general knapsack problem have a computational requirement that grows exponentially in the key size and, therefore, are impossible to implement for reasonably large key sizes. An exhaustive search is quite lengthy for large N. Suppose that N is 5. Then, the knapsack vector has five components (one corresponding to each weight), each of which could be equal to 1 (the weight is used to try to balance the knapsack) or O (the weight is not used in this try). There are 2^5 or 32 possible vectors to be tried in an exhaustive search. If N were 10, up to 1,024 tries would be covered in an exhaustive search. If N were 1,000, an exhaustive search would clearly be infeasible.
>
> Sometimes the problem is posed differently, as a cylindrical knapsack of fixed length and a set of rods of different lengths, with the problem being to find the subset of rods that will completely fill the knapsack. In either case, the problem is an example of the general class of NP problems. The "trapdoor" knapsack problem is a special case, which is not computationally difficult to solve provided that one has special information that enables the problem to be solved more easily than for the general case. In this case, the "trapdoor" enables the intended recipient who knows the secret key (the trapdoor information) to solve the knapsack problem and reveal the plaintext message without having to do an exhaustive search.

The intended receiver and originator of the public and secret keys builds a secret structure into the knapsack problem. The receiver generates the keys by first generating an "easy" knapsack vector, called a super-increasing vector, in which each weight is larger than the sum of the preceding weights. The sum of the super-increasing knapsack vector is the heaviest possible knapsack. The receiver then chooses a modulus m larger than this maximum weight and a multiplier a such that m and a are relatively prime. The "hard" knapsack vector is constructed by multiplying the "easy" vector by a, using modulus m arithmetic. The "hard' knapsack vector, arranged in order of increasing weight, forms the public (encryption) key. The "easy" vector, along with the values used for m and a, are kept as the secret key.

Merkle and Hellman's public-key system was based on special key pairs that were used to encrypt and decrypt plaintext. Briefly (see *Computer Security,* pp. 100-101), each character in the plaintext was assigned a numerical value and all the numbers were then summed together. The secret key enabled the individual numbers to be recovered and, from them, the plaintext.

[1] For the history of the trapdoor knapsack system, see *Computer. Security*, one of the *Understanding Computers* series (Alexandria, VA: Time-Life Books, 1986), pp. 100-101; Davies & Price. *Security for Computer Networks: An Introduction to Data Security in Teleprocessing and Electronic Funds Transfer, (New* York, NY: J. Wiley & Sons, 1984), p. 251; and Martin E. Hellman, "The Mathematics of Public-Key Cryptography, " *Scientific American*, vol. 241, No. 2, August 1979, p. 148.

lem of disputes between sender and receiver to be resolved through the method of digital signatures. [22]

The RSA system is based on a problem that is even older than the knapsack problem: that of "factoring" a large number (finding all the prime numbers that divide it evenly) .23 This is another computationally "one-way" problem in that factoring a large number takes much longer (by hand or by computer) than does verifying that two or more numbers are prime factors of the same large number.

The proprietary RSA system is thought to be based on the relative difficulty of finding two large prime numbers, given their product. The recipient (and originator of the key pair) randomly selects two large prime numbers, called *p* and ***q.*** These prime numbers are kept secret. Another (odd) integer e is chosen, which must pass a special mathematical test based on values of *p* and *q*. The product, *n*, of ***p*** times ***q*** and the value of e are announced as the public encryption key. Even though their product is announced publicly, the prime factors ***p*** and *q* are not readily obtained from *n*. Therefore, revealing the product of ***p*** and *q* does not compromise the secret key, which is computed from the individual values of ***p*** and ***q.***[24]

The RSA public (encrypting) key consists of two integers, ***n*** and e, where *n* is the product n=(p)(q). If each block of the plaintext message is represented as an integer between O and (n – 1), then encryption is accomplished by raising the plaintext to the eth power, modulus *n*.

The secret (signing and/or decrypting) key, *d*, is computed from ***p, q,*** and e as the "multiplicative inverse" of e, modulus (p–I)(q–1); that is, the product of d and eis1,modulus(p–l)(q–1). Thus, individual knowledge of *p* and *q* are thought to be necessary to create the secret key. Decryption is accomplished by raising the ciphertext to the *dth* power, modulus *n*.

It is possible to break the RSA cipher if the prime factors p and q can be determined by factoring the value of ***n*** that was given in the public key. Many factoring algorithms exist, some based on the work of Fermat, Legendre, and Gauss. Others have been developed more recently to take advantage of computers and special processors to do fast factorization.

Depending on the factorization method used, it is possible to estimate the number of computational steps required to factor a number, *n*. The number of steps and the speed with which they can be performed determine the time required to factor ***n.*** The number of steps required to factor ***n***—thus, the work and time required to "break" the RSA cipher by the factorization approach (finding ***p*** and q)–increases rapidly as the number of digits inn increases.[25] Thus, an important feature of the

[22] Other public-key systems have been developed, some earlier than RSA, but have not gained as wide an acceptance in commercial markets. There continue to be new developments in public-key cryptography (see, e.g., S. Goldwaaser, S. Micali, and R. Rivest, MIT Laboratory for Computer Science, "A Digital Signature Scheme Secure Against Adaptive Chosen Message Attack, Rev. Apr. 23, 1986), but some of these are of more academic interest than immediate practicability for safeguarding communications

[23] For discussions of the underlying mathematics, see Davies & Price, op. cit., ch. 8; Rivest Shamir, and Adelman in the February 1978 *Communications of the ACM,* and Hellman in the August 1979 *Scientific American* op. cit.; *Computer Security*, op. cit., pp. 112-115, has an illustrated example.

The relationship between the RSA exponential functions used to encipher and decipher follows from an identity due to Euler and Fermat which demonstrates the properties that e and d must have, givenp and *q*.

[24] Certain special values of (p)(q) can be factored easily—when *P* and *q are* nearly equal, for instance. These special cases need to be avoided in selecting suitable keys.

[25] According t. Davies & Price, op. cit., pp. 242-244, them-y shows that, for the better-known factorization algorithms, the relationship between the number of steps and *n* is exponential. In the early 1980s, experimental work doing fast factorization using a number of techniques, including special processors, pointed to a "limit" of 70 to 80 decimal digits for factoring in one day.

Advances in theory and in microprocessor and computer technologies can serve to make estimates of this type obsolete. For example, Rivest, Shamir, and Adelman's 1978 article in the *Communications of the ACM* (February 1978, p. 125) provided a table estimating the number of operations required to factor *n* using the fastest-known method then. Assuming that a computer was used and that each operation took one microsecond, the authors estimated that a 50-decimal-digit value of *n* could be factored in under 4 hours, a 75-digit value in 104 days, a 100-digit value in 74 years, and that a 200-decimal-digit *n* would require almost 4 billion years to factor.

RSA cipher is that the desired level of security (measured by the work required to break the cipher, or its "work factor") can be tailored from a wide range of levels simply by varying the number of digits in the key.

Davies and Price report on a new factorization method using parallel computation by special, large-scale integration (LSI) devices. Assuming that the parallel LSI devices use 128-bit arithmetic and run off a 1O-MHZ clock, a 100-decimal-digit value of ***n*** would take a little over 2 years to factor, a 150-digit ***n*** would take 6,300 years to factor, and a 200-digit *n* would take 860,000 years to factor.[26] Current implementations of the cipher use keys of 200 digits or longer; that is, the number n has 200 or more decimal digits.

Rivest, Shamir, and Adelman formed RSA Data Security, Inc., in 1982 and obtained an exclusive license for their invention from MIT, which owned the patent.[27] RSA Data Security has developed proprietary software packages implementing the RSA cipher on personal computer networks. These packages, being marketed commercially, provide software-based communication safeguards, including message authentication, digital signatures, key management, and encryption. Another offering is designed to safeguard data files and spreadsheets being transmitted between intelligent workstations, electronic mail networks, and files being stored locally. The RSA Data Security package that safeguards electronic mail and spreadsheets sells for about $250, including one copy of the program, a key generating program, and a registered and authenticated user identification number.

Digital Signatures

Encryption or message authentication alone can only safeguard a communication or transaction against the actions of third parties. They cannot fully protect one of the communicating parties from fraudulent actions by the other (forgery or repudiation of a message or transaction, for example) and cannot in themselves resolve contractual disputes between the two parties. Paper-based systems have long been based on mechanisms like letters of introduction for identification of the parties, signatures for authenticating a letter or contract, and sealing a (physical) envelope for privacy. The contractual value of paper documents hinges on the recognized legal validity of the signature and on the laws against forgery.

It is possible to provide equivalent functions for electronic documents by using a digital signature to authenticate the contents and also prove who originated the document (because only one party knows the secret information used to create the signature). A digital signature can be created using a symmetric cipher (such as DES), but in general asymmetric ciphers provide for more efficient operations. The digital signature method in most common use commercially is based on the RSA cipher.[28] The digital signature can be used with encipherment if privacy is required.

The equivalent of a "letter of introduction" is still necessary to verif y that the correct public key is used to check the digital signature since an adversary might try to spoof the signature system by substituting his or her own public key and signature for the real author's. This letter of introduction could be accomplished by several means. The RSA Data Security system provides "signed key server certificates" by attaching the corporation's own digital signature to the users' public keys so that users can attach their certified public signature keys to their signed messages. Note that although a public-key cipher system is used to set up the digital signature system, the actual text of the message can be sent in plaintext, if desired, or it can be encrypted using DES or the public-key cipher.[29]

The RSA Data Security digital signature system works as follows:

First, a cryptographic "hashing" algorithm creates a shorter, 128-bit "digest" of the message. The message digest, similar to a checksum, is virtually

[26] See Davies & Price, op. cit., pp. 243-244.

[27] Other university research in cryptography has also been Patented and licensed. For instance, Stanford University has four cryptography patents available for licensing on a non-exclusive basis, for a wide range of potential applications (including protection of tape and disk drives; time-shared computers; satellite, microwave, and mobile radio communications equipment; computer terminals; and electronic banking). Stanford University considers that one of these patents (Public Key Cryptographic Apparatus and Method, U.S. Patent #4,218,582, Aug. 19,1980, Martin E. Hellman and Ralph C. Merkle), covers any public-key system in any implementation.

Source: Letter dated 9/29/86 to OTA from Lisa Kuuttila, Stanford Office of Technology Licensing, Re: Stanford Dockets S77-012,-015, -048; S78-080, "Encryption Technology. "

[28] See Davies & Price, op. cit., ch. 9, for a general treatment of digital signatures and alternative methods.

[29] For example, if the RSA digital signature is used to sign and encrypt, the sender's secret key is used to sign the message and the intended recipient's public key is used to encrypt the message. The recipient uses his secret key to decrypt the message and the sender's public key to check the signature. In practice, the RSA digital signature system is used to transmit a DES key for use in encrypting the text of a message because DES can be implemented in hardware and is much faster than using the RSA algorithm to encrypt text in software.

unique[30] to each text. If a single bit of the plaintext message is altered, the message digest will change substantially. A one-way hashing function is used to prevent the document from being reconstructed from the digest.

Next, the message digest is encrypted with the author's secret key.[31] In the RSA system, each key is the inverse of the other; that is, each key can decipher text enciphered with the other key. Therefore, using Party A's public key to decipher a message into sensible plaintext proves that Party A's secret key was used to encipher the message. The integrity of this system hinges on preventing the secret key from being compromised and ensuring that an imposter does not post his own public key and pretend that it is the real Party A's.

[30] According to the tender. the probability of two different plaint exts having the same message digest is on the order of one in a trillion

[31] Note that for ordinary encryption to preserve privacy, the recipient 's public key is the one used to encrypt.

The enciphered message digest is attached to the text and both are sent to the intended recipient. The recipient removes the appended message digest and runs the text of the message through the same hashing function to produce his own copy of the message digest. Then, the recipient deciphers the message digest that was sent along with the message, using the supposed author's public key.

If the two message digests are identical, then the message did indeed come from the supposed author and the contents of the text were received exactly as sent, unless someone has learned the author's secret key and used it to forge a message digest for a message of his own or one that he has altered.

If the author wants to keep the text of the message private, so that only the intended recipient can read it, he or she can encrypt the signed message, using the recipient's public key. Then, the recipient first uses his or her own secret key to decrypt the signed message before going through the procedure described above.