Editor: Ron Vetter, North Dakota State University, IACC Bldg., Rm. 258, Fargo, ND 58105-5164; (701) 231-7084; rvetter@plains.nodak.edu

Public key cryptography

Simson L. Garfinkel

Suppose you want to buy something over the Internet with your credit card. You might simply send the 16-digit Visa number and four-digit expiration date over the Net via e-mail. But this approach has its risks: A crook might break into the company's computer and learn your credit card number.

Using the World Wide Web isn't much better. Today, some companies are inviting customers to type their credit card numbers into forms displayed by WWW clients such as Mosaic or Lynx. But when the content of those forms is sent back over the Internet to the Web server, it can be "sniffed out" by a packet-sniffer application, placing both the buyer and the seller at financial risk.

Some businesses don't worry about stolen credit card numbers—they think their bank, or credit card companies, will cover any losses that result. This is no longer the case. Credit card issuers have said that if customer accounts are compromised because a business does not adequately protect the account information, that business will be held liable for the full cost of the fraud. Others collecting credit card numbers over the Internet think they can protect themselves by merely splitting up the credit card number into two eight-digit parts for transmission in separate e-mail messages or on separate Web pages. But this approach is also problematic and will be directly targeted if such account number splitting becomes popular.

How do you make purchases over the Net when crooks armed with packet-sniffers and other tools are on the prowl for credit card numbers? The obvious solution is to use cryptography.

Cryptography offers protection

The obvious solution is to use cryptography. When sensitive information is encrypted with a powerful encryption algorithm such as DES (the US government's Data Encryption Standard), the information can be protected from eavesdroppers. DES uses a 56-bit key, meaning that someone intercepting a DES-encrypted message would have to try, on average, $(2^{56})/2$, or 3.6×10^{16}, different keys to find the correct match.

But how do you choose the correct key? To be secure, the key must be chosen randomly, and it must be changed for every encrypted message. Yet for the customer and the company to exchange encrypted messages, they must choose the same encryption key. This appears impossible without prior arrangement.

In the fall of 1975, two researchers at Stanford University were struggling with this very problem. The researchers, Whitfield Diffie and Martin Hellman, wrote a paper titled "Multi-User Cryptographic Techniques," which proposed a new kind of cryptography aimed at solving this problem. They called their system public key cryptography.

In their paper,[1] Diffie and Hellman asserted that it should be possible to create a multiuser cryptography system in which a message could be encrypted with one key and decrypted with another. The paper didn't propose a workable public key system; it simply discussed the kinds of applications such a system would allow. Written for the 1976 National Computer Conference but finished in December 1975, the paper was circulated among others working in the same field.

Diffie and Hellman realized that any workable public key encryption system would have to be based on the concept of a one-way, or trap door, function. The idea is to have a mathematical function that can easily be computed in one direction but whose inverse would be difficult or even impossible to calculate without specific information. Within a few months, they had devised a special solution to the problem. The system is not pure public key cryptography, because it requires the active participation of two individuals (or programs) to exchange a cryptographic key. Nevertheless, it has found wide use in securing communications.

A copy of the paper describing this solution[2] ended up at the Massachusetts Institute of Technology, where it inspired Ronald Rivest, Adi Shamir, and Len Adelman to develop a true public key cryptography system known today as RSA. With RSA, users can create both a public key and a secret key. Anything encrypted with the public key can be decrypted only with the secret key, and vice versa. RSA is thus ideally suited for electronic mail systems: I can send you my public key over open channels, such as the Internet. You can then use my public key to encrypt a message and send that message back to me; no one intercepting your message can decipher it, although I can read it by decrypting the message with my secret key.

The RSA algorithm can also be used for a kind of unforgeable digital signature. In this application, the secret key is used to encrypt a message, which can then itself be decrypted by anyone possessing the public key. Digital signatures can play a role in many activities that do not require secrecy but require sender authentication and guaranteed message integrity.

Delayed for 20 years

Because it offers the possibility of true privacy and secrecy, public key cryptography has become a cause célèbre among many computer users. Nevertheless, the RSA algorithm, developed in 1977, remained the stuff of academic curiosity for nearly 20 years and did not enjoy widespread use.

Three factors combined to delay the spread of public key cryptography: the speed disparity between low-cost and high-cost computers, intellectual property law, and the US government's export control laws, which remain a problem to this day.

Privacy and key length

The RSA encryption system's security rests in the difficulty of determining a user's secret key from his or her public key, which is essentially the difficulty of factoring a very large number. The longer the key, the more security it provides. When RSA was introduced to the public in the September 1977 edition of *Scientific American*, the article included a secret message encrypted with a 129-digit key. At the time, Rivest was quoted as saying that this message would be secure for 4 quadrillion years. Later he admitted making up the figure and said he had no idea how long the cipher would be secure.

Although longer keys provide more security, they unfortunately increase the time required for encryption and decryption. During the late 1970s through the end of the 1980s, speed disparity between low-cost desktop computers and high-performance supercomputers was such that any message that could be encrypted on a low-cost computer in a few minutes could be broken, or forcibly decrypted, on a supercomputer running for a year or so. The cipher was not secure after all.

Fortunately, increases in computational power favor the code maker over the code breaker: Longer keys take linearly more time to compute but exponentially more time to break. By the early 1990s, desktop computers could encrypt messages using 512-bit keys. Today, 1,024-bit keys are becoming the norm, and new encryption programs can use keys of 2,048 bits, or even longer, without significant delay.

However, the RSA key is not the only weak point of most cryptosystems. Because public key encryption is from five to 20 times slower than conventional cryptography, most practical public key systems simply use the public key algorithm to exchange a randomly chosen key that is different for every message. This per-message key is then used to encrypt the message with a traditional secret-key cipher. Although DES is still widely used for this purpose, 56 bits is no longer considered secure. Instead, Triple DES, with 168 bits, or IDEA (International Data Encryption Algo-rithm), with 128 bits, is commonly used.

Intellectual property

Widespread use of public key cryptography was further delayed by three software patents that covered the field: patent 4,200,770, which covers the Diffie-Hellman key exchange algorithm; patent 4,218,582, which covered the "Knapsack" public key system, but was interpreted to cover *all* public key cryptography; and patent 4,424,414, which covers the RSA cryptosystem. Table 1 summarizes the public key cryptography patents.

In the 1980s, these patents were licensed to two California companies: RSA Data Security and ViaCrypt. Both companies aggressively protected their patent rights, effectively preventing the algorithms from being used in the US without a license. The added complexity and transaction costs associated with the licenses was a major stumbling block for companies and individuals seeking to use the technology.

In recent years, the patent landscape has changed slightly. Today, RSA Data Security distributes an RSA "reference implementation" that can be used freely in noncommercial products. And with the patents' impending expiration (see last column of Table 1), the patent holders have been more willing to license their patents on reasonable terms. Furthermore, there are now several lawsuits challenging the validity of the patents themselves.

Table 1. Public key cryptography patents.*

Number and title	Covers invention	Inventors	Assignee	Date filed	Date granted	Date expires
4,200,770, Cryptographic Apparatus and Method	Diffie-Hellman key exchange	Martin E. Hellman, Bailey W. Diffie, Ralph C. Merkle	Stanford University	Sept. 6, 1977	Apr. 29, 1980	Apr. 29, 1997
4,218,582 Public Key Cryptographic Apparatus and Method	Knapsack and possibly all public key cryptography	Martin E. Hellman, Ralph C. Merkle	Stanford University	Oct. 6, 1977	Aug. 19, 1980	Aug. 19, 1997
4,405,829 Cryptographic Communications System and Method	RSA encryption	Ronald L. Rivest, Adi Shamir, Leonard M. Adelman	Massachusetts Institute of Technology	Dec. 14, 1977	Sept. 20, 1983	Sept. 20, 2000

*Source: *Practical Unix and Internet Security*, by Simson L. Garfinkel and Gene Spafford, O'Reilly & Associates, 1996.

Export restrictions

The third stumbling block continues to be the US government's restrictions on cryptography export. Because other countries can use such programs to hide their communications during hostilities, the US government classifies encryption software as munitions. American companies are largely prohibited from selling to overseas customers any programs that include strong cryptographic features.

These export restrictions have limited the availability of domestic encryption as well. Encryption software developers find it too expensive to create two versions of their programs—one with strong cryptography for domestic use and another with cryptography weak enough for export. So in the US, most developers sell only the weaker cryptography software.

Even distributing programs intended only for domestic use entails the risk of prosecution if the program should escape overseas. A US federal prosecutor investigated Philip R. Zimmermann for more than two years after his program PGP (Pretty Good Privacy) was released on the Internet. Although charges were never filed, Zimmermann now faces tens of thousands of dollars in legal fees. That is a powerful deterrent for others thinking of making cryptography software freely available.

The US software industry says these export restrictions are costing the US its lead in portions of the overseas software market. For this reason, two bills have been introduced in Congress that would repeal the restrictions. The IEEE, the Association for Computing Machinery, and numerous corporations have supported this legislation.

Cryptography doesn't always mean privacy

Encryption advocates frequently use the terms cryptography and privacy interchangeably. One of the best examples of this usage is the new Internet packet-level security specifications, IPsec, which claim to use cryptography to provide privacy.

It's important to remember that cryptographic privacy is not unconditional: It protects information only while it is in transit, and it does not protect information from someone who has the key to the encrypted message or who has discovered an error in the cryptographic algorithm or protocol.

Indeed, while cryptography can protect e-mail and financial information from interception, it can't protect the information from disclosure by a party who has access to it—whether the disclosure is intentional or accidental. Consider a fictitious example in which Bob sends e-mail to Janice about their mutual co-worker, Theresa. Janice replies to the message, but instead of putting Theresa's name in the *Subject:* field, she accidentally puts it in the *Cc:* field. If Janice is using a strong encryption system like PGP, the program

COMMONLY USED ENCRYPTION ALGORITHMS

DES—The Data Encryption Standard, an encryption algorithm developed in the 1970s by the National Bureau of Standards (since renamed the National Institute of Standards and Technology, or NIST) and IBM. DES uses a 56-bit key.

RC2—A block cipher originally developed by Ronald Rivest and claimed as a trade secret by RSA Data Security. This algorithm was revealed by an anonymous Usenet posting in 1996 and appears to be reasonably strong, although particular keys are weak. RC2 is sold with an implementation that allows key lengths between 1 and 2,048 bits. The RC2 key length is often limited to 40 bits in software sold for export.

RC4—A stream cipher originally developed by Ronald Rivest and kept as a trade secret by RSA Data Security. This algorithm was revealed by an anonymous Usenet posting in 1994 and appears to be reasonably strong, although particular keys are weak. RC4 is sold with an implementation that allows key lengths between 1 and 2,048 bits. The RC4 key length is often limited to 40 bits in software sold for export.

RC5—A block cipher developed by Ronald Rivest and published in 1994. RC5 allows a user-defined key length, data block size, and number of encryption rounds. RSA Data Security is working to have RC5 included in numerous Internet standards, including IPsec.

IDEA—The International Data Encryption Algorithm, developed in Zurich, Switzerland, by James L. Massey and Xuejia Lai and published in 1990. IDEA uses a 128-bit key and is believed to be quite strong. It is used by the popular program PGP (described in an accompanying sidebar) to encrypt files and electronic mail. Unfortunately, wider use of IDEA may be hampered by a series of patents on the algorithm, which are currently held by Ascom-Tech AG in Solothurn, Switzerland. Ascom-Tech supposedly will allow IDEA to be used royalty free in implementations of PGP outside the US, but concerned users should verify the terms with Ascom-Tech or their licensees directly.

Blowfish—An extremely fast encryption algorithm developed by Bruce Schneier. Blowfish has a 128-bit key. Its security is unknown.

Skipjack—A classified (secret) algorithm developed by the National Security Agency. Reportedly, a top secret security clearance is required to see the algorithm's source code and design specifications. Skipjack uses an 80-bit key and is the algorithm used by the Clipper encryption chip.

Source: *Practical Unix and Internet Security*, by Simpson L. Garfinkel and Gene Spafford, O'Reilly & Associates, 1996.

Products that use cryptography

AT&T Telephone Security Device 3600—A voice encryption device for conventional analog phones. The TSD 3600 uses the Diffie-Hellman algorithm to exchange a session key, 16 bits of which are displayed on a four-digit display. The stream encryption cipher can be either DES, Skipjack, or a proprietary algorithm.

PGP (Pretty Good Privacy)—A widely available encryption and signature program. PGP uses the IDEA algorithm to encrypt mail messages with a randomly chosen session key, which in turn is encrypted with the RSA system.

PGPfone—A voice encryption system available for PCs running Windows 95 and for Macintosh computers. PGPfone uses the Diffie-Hellman algorithm to exchange a session key, and then it can use either triple DES, IDEA, or Blowfish to encrypt data.

RSA Secure—A hard-disk encryption utility sold by RSA Data Security. RSA Secure uses RC4 with a 128-bit key to encrypt data stored on a hard disk. The decryption key can be split into several parts and stored on floppy disks for emergency access in case the operator forgets the password. A thresholding system is used so that four key splits can be created, but only three are needed to recover the data.

Netscape Navigator—The Netscape Navigator implements both SSL (Netscape's Secure Socket Layer) and SHTTP (Secure Hyper-Text Transfer Protocol). Netscape uses the RSA encryption algorithm to exchange keys and digital signatures. Data is encrypted with RC4, with a key length of either 128 bits (domestic version) or 40 bits (export version).

will happily sign Janice's message with her secret key, then encrypt the message for both Bob and Theresa with their public keys, and send it along. The contents of the communication between Bob and Janice will be completely private—until Theresa opens her mailbox!

References

1. W. Diffie and M.E. Hellman, "New Directions in Cryptography," *IEEE Trans. Information Theory*, Nov. 1976, pp. 644-654.
2. W. Diffie and M.E. Hellman, "Multi-User Cryptographic Techniques," *Proc. AFIPS Nat'l Computer Conf.*, 1976, pp. 109-112.

Simson L. Garfinkel *is a freelance writer and coauthor, with Gene Spafford, of* Practical Unix and Internet Security, *O'Reilly & Associates, 1996. He can be reached at simsong@vineyard.net.*

New Products

Contact or send releases to Scott Hamilton, *Computer*, 10662 Los Vaqueros Circle, PO Box 3014, Los Alamitos, CA 90720-1314; s.hamilton@computer.org

Software development tools

Rational Apex Cross

Rational Software Corporation announced Rational Apex C/C++ Cross, an integrated software-engineering environment for designing, developing, and maintaining embedded and real-time C/C++ projects. This fully integrated system includes a production-quality C and C++ cross-compiler, a target-oriented source-level debugger, a C/C++ code-rule checker, and a program browser. The environment provides powerful configuration-management and version-control capabilities. The initial release offers support for Motorola's PowerPC microprocessors, with support for other Motorola microprocessors planned for later this year. It is available for Sun Sparc systems running Solaris, with additional platform support being added throughout the year. Prices start at $14,000.

Reader Service Number 36

Expert C++

CenterLine Software introduced its C++Expert error-detection tool. The tool combines the company's On-line Advice for C++ and its SourceWise three-way error-detection technology to help developers enhance code quality and improve their C++ fluency and understanding.

On-line Advice for C++ automatically checks code using expert rules derived from Scott Meyers' *More Effective C++* (Addison-Wesley, 1996). At compile time, it scans source code for usage patterns that can jeopardize code quality and notifies the user. Users can then explore detailed explanations by clicking on the flagged line of code.

The SourceWise automatic error detection checks code during compile, link, and runtime, providing in-depth error detection earlier in the development cycle. At compile time, the tool detects problems such as dead code and unused variables. At link time, it detects multiple definition errors across files. At runtime, it tracks memory usage in heap, stack, and static memory, as well as language-specific errors such as illegal casts and null references.

The package is available for SunOS and Solaris on Sparc and UltraSparc, with support for additional platforms later this year. It costs $995 per seat.

Reader Service Number 37

Embeddable C++

NewCode Technology Inc. introduced *NCi*, the company's standards-compliant C++ interpreter designed to be embedded in end-user applications. The interpreter becomes a runtime component of an applications, letting end-users dynamically extend functionality in their C++ products. The package lets interpreted code call compiled