International Journal of Application or Innovation in Engineering & Management (IJAIEM)
Web Site: www.ijaiem.org Email: editor@ijaiem.org, editorijaiem@gmail.com
Volume 2, Issue 3, March 2013 ISSN 2319 - 4847

# CRYPTOGRAPHY AS AN INSTRUMENT TO NETWORK SECURITY

**T. P.Wasnik[1], Vishal S. Patil[2], Sushant A. Patinge[3], Sachin R. Dave[4], Gaurav J. Sayasikamal[5]**

ME (CSE) Scholar, Department of CSE,

P R Patil College of Engineering & Technology Amravati, Amravati-444605, India.

## ABSTRACT

***Network security is a complicated subject, historically only tackled by well-trained and experienced experts. However, as more and more people become ``wired'', an increasing number of people need to understand the basics of security in a networked world. This document was written with the basic computer user and information systems manager in mind, explaining the concepts needed to read through the hype in the marketplace and understand risks and how to deal with them. Some history of networking is included, as well as an introduction to TCP/IP and internetworking. We can consider risk management, network threats, firewalls, and more special-purpose secure networking devices. This is not intended to be a ``frequently asked questions'' reference, nor is it a ``hands-on'' document describing how to accomplish specific functionality. It is hoped that the reader will have a wider perspective on security in general, and better understand how to reduce and manage risk personally, at home, and in the workplace. Cryptography and Network Security Does security provide some very basic protections that we are naive to believe that we don't need? During this time when the Internet provides essential communication between tens of millions of people and is being increasingly used as a tool for commerce, security becomes a tremendously important issue to deal with. There are many aspects to security and many applications, Ranging from secure commerce and payments to private Communications and protecting passwords. One essential aspect for secure communications is that of cryptography. Cryptography is the science of writing in secret code and is an ancient art. The first documented use of cryptography in writing dates back to circa 1900 B.C. when an Egyptian scribe used non-standard hieroglyphs in an inscription. In data and telecommunications, cryptography is necessary when communicating over any untrusted medium, which includes just about any network, particularly the Internet. Within the context of any application-to-application communication, there are some specific security requirements, including:***

***Authentication: The process of proving one's identity. (The primary forms of host-to-host authentication on the Internet today are name-based or address-based, both of which are notoriously weak.)***
***Privacy/confidentiality: Ensuring that no one can read the message except the intended receiver.***
***Integrity: Assuring the receiver that the received message has not been altered in any way from the original.***
***Non-repudiation: A mechanism to prove that the sender really sent this message. Cryptography, then, not only protects data from theft or alteration, but can also be used for user authentication.***
***The three types of cryptographic algorithms that will be discussed are***
***Secret Key Cryptography (SKC): Uses a single key for both encryption and decryption.***
***Public Key Cryptography (PKC): Uses one key for encryption and another for decryption***
***Hash Functions: Uses a mathematical transformation to irreversibly "encrypt" information***

**Keywords:** Cryptography, SKC, PKC, Authentication, Confidentiality

## 1. INTRODUCTION

Cryptography has had an interesting history and has undergone many changes through the centuries. It seems that keeping secrets has been important throughout the ages of civilization for one reason or another. Keeping secrets gives individuals or groups the ability to hide true intentions, gain a competitive edge, and reduce vulnerability. The changes that cryptograph has undergone throughout history closely follow the advances in technology. Cryptography methods began with a person carving messages into wood or stone, which were then passed to the intended individual who had the necessary means to decipher the messages. This is a long way from how cryptography is being used today. Cryptography that used to be carved into materials is now being inserted into streams of binary code that passes over network wires, Internet communication paths, and airwaves. Cryptography has roots that began around 2000 B.C. in Egypt when hieroglyphics were used to decorate tombs to tell the story of the life of the deceased. The practice was not as much to hide the messages themselves, but to make them seem more noble, ceremonial, and majestic. Around 400 B.C., the Spartans used a system of encrypting information by writing a message on a sheet of papyrus, which was

wrapped around a staff. The message was only readable if it was around the correct staff, which allowed the letters to properly match up. This is referred to as the scytale cipher when the papyrus was removed from the staff; the writing appeared as just a bunch of random characters.

In the past, messengers were used as the transmission mechanism, and encryption helped protect the message in case the messenger was captured. Today, the transmission mechanism has changed from human beings to packets carrying 0's and 1's passing through network cables or open airwaves. The messages are still encrypted in case an intruder captures the transmission mechanism (the packets) as they travel along their paths.

As computers came to be, the possibilities for encryption methods and devices advanced, and cryptography efforts expanded exponentially. This era brought unprecedented opportunity for cryptographic designers and encryption techniques. The most well-known and successful project was Lucifer, which was developed at IBM. Lucifer introduced complex mathematical equations and functions that were later adopted and modified by the U.S. National Security Agency (NSA) to come up with the U.S. Data Encryption Standard (DES). DES has been adopted as a federal government standard, is used worldwide for financial transactions, and is imbedded into numerous commercial applications. DES has had a rich history in computer-oriented encryption and has been in use for over 20 years. Cryptography has had its days in the political limelight with governments enforcing transborder restrictions and hindering the use of cryptography in certain sectors by imposing export regulations. Law enforcement developed their own encryption chip, the Clipper Chip, to decipher communication that had to do with suspected criminal activity and drug movements, which has raised many questions about the public's privacy versus the government's right to eavesdrop. A majority of the protocols developed at the dawn of the computing age have had upgraded to include cryptography to add necessary layers of protection. Encryption is used in hardware devices and software to protect data, banking transactions, corporate extranets, e-mail, Web transactions, wireless communication, storing of confidential information, faxes, and phone calls.

## 2. DEFINITION

Encryption is a method of transforming original data, called ***plaintext*** or ***cleartext***, into a form that appears to be random and unreadable, which is called ***ciphertext***. Plaintext is either in a form that can be understood by a person (a document) or by a computer (executable code). Once it is not transformed into ciphertext, human nor can machine properly process it until it is decrypted. This enables the transmission of confidential information over insecure channels without unauthorized disclosure. When data is stored on a computer, it is usually protected by logical and physical access controls.


**Figure 1**

A system that provides encryption and decryption is referred to as a *cryptosystem* and can be created through hardware components or program code in an application. The cryptosystem uses an encryption algorithm, which determines how simple or complex the process will be. Most algorithms are complex mathematical formulas that are applied in a specific sequence to the plaintext. Most encryption methods use a secret value called a key usually a long string of bits, which works with the algorithm to encrypt and decrypt the text.

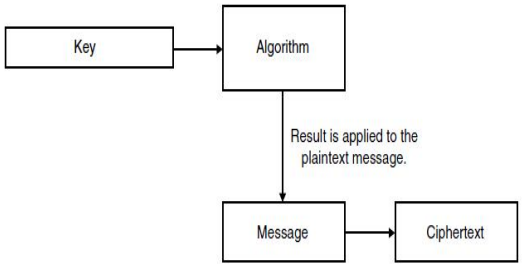


**Figure 2**

The algorithm, the set of mathematical rules, dictates how enciphering and deciphering take place. Many algorithms are publicly known and are not the secret part of the encryption process. The way that encryption algorithms work can be kept secret from the public, but many of them are publicly known and well understood. If the internal mechanisms of the algorithm are not a secret, then something must be. The secret piece of using a well-known encryption algorithm is the key. The key can be any value that is made up of a large sequence of random bits. Is it just any random number of bits crammed together. An algorithm contains a keyspace, which is a range of values that can be used to construct a key. The key is made up of random values within the keyspace range. The larger the keyspace, the more available values can be used to represent different keys, and the more random the keys are, the harder it is for intruders to figure them out. A large keyspace allows for more possible keys. The encryption algorithm should use the entire keyspace and choose the values to make up the keys as random as possible. If a smaller keyspace were used, there would be fewer values to choose from when forming a key, this would increase an attacker's chance of figuring out the key value and deciphering the protected information.

If an eavesdropper captures a message as it passes between two people, she can view the message, but it appears in its encrypted form and is therefore unusable. Even if this attacker knows the algorithm that the two people are using to encrypt and decrypt their information, without the key, this information remains useless to the eavesdropper.

## 3. THE PURPOSE OF CRYPTOGRAPHY

The main use of cryptography is to provide the following as mentioned earlier.

(1) privacy or confidentiality

(2) data integrity

(3) authentication and

(4) non-repudiation.

1. ***Confidentiality*** is a service used to keep the content of information from all but those authorized to posses it. Secrecy is a term synonymous with confidentiality and privacy. There are numerous approaches to providing confidentiality, ranging from physical protection to mathematical algorithms which render data unintelligible.

2. ***Data integrity*** is a service which addresses the unauthorized alteration of data. To assure data integrity, one must have the ability to detect data manipulation by unauthorized parties. Data manipulation includes such things as insertion, deletion, and substitution.

3. ***Authentication*** is a service related to identification. This function applies to both entities and information itself. Two parties entering into a communication should identify each other. Information delivered over a channel should be authenticated as to origin, date of origin, data content, time sent, etc. For these reasons this aspect of cryptography is usually subdivided into two major classes: entity-authentication and data-origin authentication. Data origin authentication implicitly provides data integrity.

4. ***Non-repudiation*** is a service which prevents an entity from denying previous commitments or actions. When disputes arise due to an entity denying that certain actions were taken, a means to resolve the situation is necessary.

Cryptography, then, not only protects data from theft or alteration, but can also be used for user authentication. There are, in general, three types of cryptographic schemes typically used to accomplish these goals: secret key (or symmetric) cryptography, public-key (or asymmetric) cryptography, and hash functions. In all cases, the initial unencrypted data is referred to as plaintext. It is encrypted into ciphertext, which will in turn usually be decrypted into usable plaintext.

## 4. TYPES OF CRYPTOGRAPHIC ALGORITHMS

There are several ways of classifying cryptographic algorithms. For purposes of this paper, they will be categorized based on the number of keys that are employed for encryption and decryption, and further defined by their application and use. The three types of algorithms that will be discussed are:

1. Secret Key Cryptography (SKC): Uses a single key for both encryption and decryption

2. Public Key Cryptography (PKC): Uses one key for encryption and another for decryption

Hash Functions: Uses a mathematical transformation to irreversibly "encrypt" information

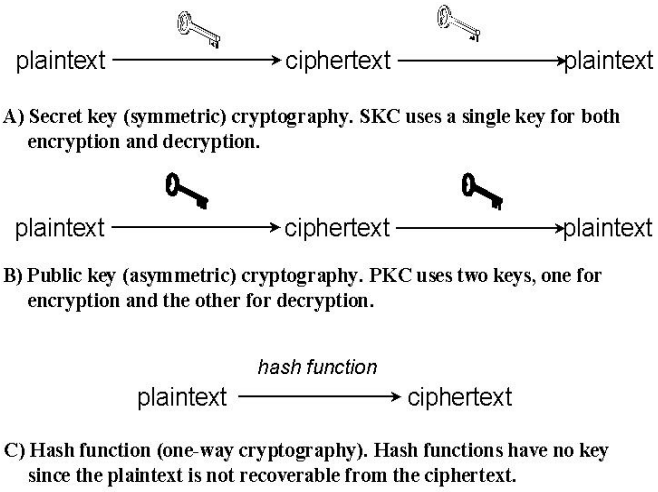


**Figure 3**

### *A. SECRET KEY CRYPTOGRAPHY*

With secret key cryptography, a single key is used for both encryption and decryption. As shown in Figure 3A, the sender uses the key (or some set of rules) to encrypt the plaintext and sends the ciphertext to the receiver. The receiver applies the same key (or ruleset) to decrypt the message and recover the plaintext. Because a single key is used for both functions, secret key cryptography is also called symmetric encryption. With this form of cryptography, it is obvious that the key must be known to both the sender and the receiver; that, in fact, is the secret. The biggest difficulty with this approach, of course, is the distribution of the key. Secret key cryptography schemes are generally categorized as being either stream ciphers or block ciphers. Stream ciphers operate on a single bit (byte or computer word) at a time and implement some form of feedback mechanism so that the key is constantly changing. A block cipher is so called because the scheme encrypts one block of data at a time using the same key on each block. In general, the same plaintext block will always encrypt to the same ciphertext when using the same key in a block cipher whereas the same plaintext will encrypt to different ciphertext in a stream cipher. Stream ciphers come in several flavors but two are worth mentioning here. Self-synchronizing stream ciphers calculate each bit in the keystream as a function of the previous n bits in the keystream. It is termed "self-synchronizing" because the decryption process can stay synchronized with the encryption process merely by knowing how far into the n-bit keystream it is. One problem is error propagation; a garbled bit in transmission will result in n garbled bits at the receiving side. Synchronous stream ciphers generate the keystream in a fashion independent of the message stream but by using the same keystream generation function at sender and receiver. While stream ciphers do not propagate transmission errors, they are, by their nature, periodic so that the keystream will eventually repeat. Block ciphers can operate in one of several modes;

The following four are the most important:

- *Electronic Codebook (ECB) mode* is the simplest, most obvious application: the secret key is used to encrypt the plaintext block to form a ciphertext block. Two identical plaintext blocks, then, will always generate the same ciphertext block. Although this is the most common mode of block ciphers, it is susceptible to a variety of brute-force attacks.

- *Cipher Block Chaining (CBC) mode* adds a feedback mechanism to the encryption scheme. In CBC, the plaintext is exclusively-ORed (XORed) with the previous ciphertext block prior to encryption. In this mode, two identical blocks of plaintext never encrypt to the same ciphertext.

- *Cipher Feedback (CFB) mode* is a block cipher implementation as a self-synchronizing stream cipher. CFB mode allows data to be encrypted in units smaller than the block size, which might be useful in some applications such as encrypting interactive terminal input. If we were using 1byte CFB mode, for example, each incoming character is placed into a shift register the same size as the block, encrypted, and the block transmitted. At the receiving side, the ciphertext is decrypted and the extra bits in the block i.e., everything above and beyond the one byte are discarded.

- *Output Feedback (OFB) mode* is a block cipher implementation conceptually similar to a synchronous stream cipher. OFB prevents the same plaintext block from generating the same ciphertext block by using an internal feedback mechanism that is independent of both the plaintext and ciphertext bitstreams.

Secret key cryptography algorithms that are in use today include:

1. *Data Encryption Standard (DES):* The most common SKC scheme used today, DES was designed by IBM in the 1970s and adopted by the National Bureau of Standards (NBS) in 1977 for commercial and unclassified government applications. DES is a block-cipher employing a 56-bit key that operates on 64-bit blocks. DES has a complex set of rules and transformations that were designed specifically to yield fast hardware implementations and slow software implementations, although this latter point is becoming less significant today since the speed of computer processors is several orders of magnitude faster today than twenty years ago. IBM also proposed a 112-bit key for DES, which was rejected at the time by the government; the use of 112-bit keys was considered in the 1990s, however, conversion was never seriously considered. DES is defined in American National Standard X3.92 and three Federal Information Processing Standards (FIPS)

Two important variants that strengthen DES are:

- *Triple-DES (3DES):* A variant of DES that employs up to three 56-bit keys and makes three encryption/decryption passes over the block; 3DES is also described in FIPS 46-3 and is the recommended replacement to DES.

- *DESX:* A variant devised by Ron Rivest. By combining 64 additional key bits to the plaintext prior to encryption, effectively increases the keylength to 120 bits.

2. *Advanced Encryption Standard (AES):* In 1997, NIST initiated a very public, 4-1/2 year process to develop a new secure cryptosystem for U.S. government applications. The result, the Advanced Encryption Standard, became the official successor to DES in December 2001. AES uses an SKC scheme called Rijndael, a block cipher designed by Belgian cryptographers Joan Daemen and Vincent Rijmen. The algorithm can use a variable block length and key length; the latest specification allowed any combination of keys lengths of 128, 192, or 256 bits and blocks of length 128, 192, or 256 bits. NIST initially selected Rijndael in October 2000 and formal adoption as the AES standard came in December 2001. FIPS PUB 197 describes a 128-bit block cipher employing a 128-, 192-, or 256-bit key.

The AES process and Rijndael algorithm.

- *CAST-128/256:* CAST-128, described in Request for Comments (RFC) 2144, is a DES-like substitution-permutation crypto algorithm, employing a 128-bit key operating on a 64-bit block. CAST-256 is an extension of CAST-128, using a 128-bit block size and a variable length (128, 160, 192, 224, or 256 bit) key. CAST is named for its developers, Carlisle Adams and Stafford Tavares and is available internationally. CAST-256 was one of the Round 1 algorithms in the AES process.

- *International Data Encryption Algorithm (IDEA):* Secret-key cryptosystem written by Xuejia Lai and James Massey, in 1992 and patented by Ascom; a 64-bit SKC block cipher using a 128-bit key.

- *Rivest Ciphers (*aka *Ron's Code):* Named for Ron Rivest, a series of SKC algorithms.

*RC1:* Designed on paper but never implemented.

*RC2:* A 64-bit block cipher using variable-sized keys designed to replace DES. It's code has not been made public although many companies have licensed RC2 for use in their products.

*RC3:* Found to be breakable during development.

*RC4:* A stream cipher using variable-sized keys; it is widely used in commercial cryptography products, although it can only be exported using keys that are 40 bits or less in length.

*RC5:* A block-cipher supporting a variety of block sizes, key sizes, and number of encryption passes over the data.

*RC6:* An improvement over RC5, RC6 was one of the AES Round 2 algorithms.

- *Blowfish:* A symmetric 64-bit block cipher invented by Bruce Schneier; optimized for 32-bit processors with large data caches, it is significantly faster than DES on a Pentium/PowerPC-class machine. Key lengths can vary from

32 to 448 bits in length. Blowfish, available freely and intended as a substitute for DES or IDEA, is in use in over 80 products.

- *Twofish:* A 128-bit block cipher using 128-, 192-, or 256-bit keys. Designed to be highly secure and highly flexible, well-suited for large microprocessors, 8-bit smart card microprocessors, and dedicated hardware. Designed by a team led by Bruce Schneier and was one of the Round 2 algorithms in the AES process.

- *Camellia:* A secret-key, block-cipher crypto algorithm developed jointly by Nippon Telegraph and Telephone (NTT) Corp. and Mitsubishi Electric Corporation (MEC) in 2000. Camellia has some characteristics in common with AES: a 128-bit block size, support for 128-, 192-, and 256-bit key lengths, and suitability for both software and hardware implementations on common 32-bit processors as well as 8-bit processors e.g., smart cards, cryptographic hardware, and embedded systems.

- *MISTY1:* Developed at Mitsubishi Electric Corp., a block cipher using 128-bit key and 64-bit blocks, and a variable number of rounds. Designed for hardware and software implementations, and is resistant to differential and linear cryptanalysis.

- *Secure and Fast Encryption Routine (SAFER):* Secret-key crypto scheme designed for implementation in software. Versions have been defined for 40-, 64-, and 128-bit keys.

- *KASUMI:* A block cipher using a 128-bit key that is part of the Third-Generation Partnership Project (3gpp), formerly known as the Universal Mobile Telecommunications System (UMTS). KASUMI is the intended confidentiality and integrity algorithm for both message content and signaling data for emerging mobile communications systems.

- *SEED:* A block cipher using 128-bit blocks and 128-bit keys. Developed by the Korea Information Security Agency (KISA) and adopted as a national standard encryption algorithm in South Korea.

- *Skipjack:* SKC scheme proposed for Capstone. Although the details of the algorithm were never made public, Skipjack was a block cipher using an 80-bit key and 32 iteration cycles per 64-bit block.

### *B. PUBLIC-KEY CRYPTOGRAPHY*

Public-key cryptography has been said to be the most significant new development in cryptography in the last 300-400 years. Modern PKC was first described publicly by Stanford University professor Martin Hellman and graduate student Whitfield Diffie in 1976. Their paper described a two-key crypto system in which two parties could engage in a secure communication over a non-secure communications channel without having to share a secret key. PKC depends upon the existence of so-called one-way functions, or mathematical functions that are easy to computer whereas their inverse function is relatively difficult to compute. Two simple examples:

1. *Multiplication vs. factorization:* Suppose that you have two numbers, 9 and 16, and you have to calculate the product; it should take almost no time to calculate the product, 144. Suppose instead that you have a number, 144, and you have been told which pair of integers should be multiplied together to obtain that number. You will eventually come up with the solution but whereas calculating the product took milliseconds, factoring will take longer because you first need to find the 8 pair of integer factors and then determine which one is the correct pair.

2. *Exponentiation vs. logarithms:* Suppose, you want to take the number 3 to the $6^{th}$ power; again, it is easy to calculate $3^6=729$. But if you have the number 729 and asked two integers that can be used, $x$ and $y$ so that $\log_x 729 = y$, it will take you longer to find all possible solutions and select the pair that can be used.

While the examples above are trivial, they do represent two of the functional pairs that are used with PKC; namely, the ease of multiplication and exponentiation versus the relative difficulty of factoring and calculating logarithms, respectively. The mathematical "trick" in PKC is to find a *trap door* in the one-way function so that the inverse calculation becomes easy given knowledge of some item of information. Generic PKC employs two keys that are mathematically related although knowledge of one key does not allow someone to easily determine the other key. One key is used to encrypt the plaintext and the other key is used to decrypt the ciphertext. The important point here is that it does not matter which key is applied first, but that both keys are required for the process to work. Because a pair of keys are required, this approach is also called *asymmetric cryptography*. In PKC, one of the keys is designated the *public key* and may be advertised as widely as the owner wants. The other key is designated the *private key* and is never revealed to another party. It is straight forward to send messages under this scheme. Suppose Alice wants to send Bob a message. Alice encrypts some information using Bob's public key; Bob decrypts the ciphertext using his private key.

This method could be also used to prove who sent a message; Alice, for example, could encrypt some plaintext with her private key; when Bob decrypts using Alice's public key, he knows that Alice sent the message and Alice cannot deny having sent the message (*non-repudiation*).

Public-key cryptography algorithms that are in use today for key exchange or digital signatures include:

- *RSA:* The first, and still most common, PKC implementation, named for the three MIT mathematicians who developed it *Ronald Rivest, Adi Shamir*, and *Leonard Adleman*. RSA today is used in hundreds of software products and can be used for key exchange, digital signatures, or encryption of small blocks of data. RSA uses a variable size encryption block and a variable size key. The key-pair is derived from a very large number, *n*, that is the product of two prime numbers chosen according to special rules; these primes may be 100 or more digits in length each, yielding an *n* with roughly twice as many digits as the prime factors. The public key information includes *n* and a derivative of one of the factors of *n*; an attacker cannot determine the prime factors of *n* and, therefore, the private key from this information alone and that is what makes the RSA algorithm so secure. Some descriptions of PKC erroneously state that RSA's safety is due to the difficulty in factoring large prime numbers. In fact, large prime numbers, like small prime numbers, only have two factors. The ability for computers to factor large numbers, and therefore attack schemes such as RSA, is rapidly improving and systems today can find the prime factors of numbers with more than 200 digits. Nevertheless, if a large number is created from two prime factors that are roughly the same size, there is no known factorization algorithm that will solve the problem in a reasonable amount of time; a 2005 test to factor a 200-digit number took 1.5 years and over 50 years of compute time. Regardless, one presumed protection of RSA is that users can easily increase the key size to always stay ahead of the computer processing curve. As an aside, the patent for RSA expired in September 2000 which does not appear to have affected RSA's popularity one way or the other.

- *Diffie-Hellman:* After the RSA algorithm was published, Diffie and Hellman came up with their own algorithm. D-H is used for secret-key key exchange only, and not for authentication or digital signatures.

- *Digital Signature Algorithm (DSA):* The algorithm specified in NIST's Digital Signature Standard (DSS), provides digital signature capability for the authentication of messages.

- *ElGamal:* Designed by Taher Elgamal, a PKC system similar to Diffie-Hellman and used for key exchange.

- *Elliptic Curve Cryptography (ECC):* A PKC algorithm based upon elliptic curves. ECC can offer levels of security with small keys comparable to RSA and other PKC methods. It was designed for devices with limited compute power and/or memory, such as smartcards and PDAs.

- *Public-Key Cryptography Standards (PKCS):* A set of interoperable standards and guidelines for public-key cryptography, designed by RSA Data Security Inc.

PKCS #1: RSA Cryptography Standard

PKCS #2: Incorporated into PKCS #1.

PKCS #3: Diffie-Hellman Key-Agreement Standard

PKCS #4: Incorporated into PKCS #1.

PKCS #5: Password-Based Cryptography Standard

PKCS #6: Extended-Certificate Syntax Standard

PKCS #7: Cryptographic Message Syntax Standard

PKCS #8: Private-Key Information Syntax Standard

PKCS #9: Selected Attribute Types

PKCS #10: Certification Request Syntax Standard

PKCS #11: Cryptographic Token Interface Standard

PKCS #12: Personal Information Exchange Syntax Standard

PKCS #13: Elliptic Curve Cryptography Standard

PKCS #14: Pseudorandom Number Generation Standard is no longer available

PKCS #15: Cryptographic Token Information Format Standard

- *Cramer-Shoup:* A public-key cryptosystem proposed by R. Cramer and V. Shoup of IBM in 1998.
- *Key Exchange Algorithm (KEA):* A variation on Diffie-Hellman; proposed as the key exchange method for Capstone.
- *LUC:* A public-key cryptosystem designed by P.J. Smith and based on Lucas sequences. Can be used for encryption and signatures, using integer factoring.

### *3. HASH FUNCTIONS*

*Hash functions*, also called message digests and one-way encryption, are algorithms that, in some sense, use no key (Figure 3C). Instead, a fixed-length hash value is computed based upon the plaintext that makes it impossible for either the contents or length of the plaintext to be recovered. Hash algorithms are typically used to provide a digital fingerprint of a file's contents often used to ensure that the file has not been altered by an intruder or virus. Hash functions are also commonly employed by many operating systems to encrypt passwords. Hash functions, then, provide a measure of the integrity of a file. Hash algorithms that are in common use today include:

- *Message Digest (MD) algorithms:* A series of byte-oriented algorithms that produce a 128-bit hash value from an arbitrary-length message.
- *MD2 (RFC 1319):* Designed for systems with limited memory, such as smart cards.
- *MD4 (RFC 1320):* Developed by Rivest, similar to MD2 but designed specifically for fast processing in software.
- *MD5 (RFC 1321):* Also developed by Rivest after potential weaknesses were reported in MD4; this scheme is similar to MD4 but is slower because more manipulation is made to the original data. MD5 has been implemented in a large number of products although several weaknesses in the algorithm were demonstrated by German cryptographer Hans Dobbertin in 1996.
- *Secure Hash Algorithm (SHA):* Algorithm for NIST's Secure Hash Standard (SHS). SHA-1 produces a 160-bit hash value and was originally published as FIPS 180-1 and RFC 3174. FIPS 180-2 describes five algorithms in the SHS: SHA-1 plus SHA-224, SHA-256, SHA-384, and SHA-512 which can produce hash values that are 224, 256, 384, or 512 bits in length, respectively. SHA-224, -256, -384, and -52 are also described in RFC 4634.
- *RIPEMD:* A series of message digests that initially came from the RIPE (RACE Integrity Primitives Evaluation) project. RIPEMD-160 was designed by Hans Dobbertin, Antoon Bosselaers, and Bart Preneel, and optimized for 32-bit processors to replace the then-current 128-bit hash functions. Other versions include RIPEMD-256, RIPEMD-320, and RIPEMD-128.
- *HAVAL (HAsh of VAriable Length):* Designed by Y. Zheng, J. Pieprzyk and J. Seberry, a hash algorithm with many levels of security. HAVAL can create hash values that are 128, 160, 192, 224, or 256 bits in length.
- *Whirlpool:* A relatively new hash function, designed by V. Rijmen and P.S.L.M. Barreto. Whirlpool operates on messages less than 2256 bits in length, and produces a message digest of 512 bits. The design of this has function is very different than that of MD5 and SHA-1, making it immune to the same attacks as on those hashes.
- *Tiger:* Designed by Ross Anderson and Eli Biham, Tiger is designed to be secure, run efficiently on 64-bit processors, and easily replace MD4, MD5, SHA and SHA-1 in other applications. Tiger/192 produces a 192-bit output and is compatible with 64-bit architectures; Tiger/128 and Tiger/160 produce the first 128 and 160 bits, respectively, to provide compatibility with the other hash functions mentioned above.

Hash functions are sometimes misunderstood and some sources claim that no two files can have the same hash value. This is, in fact, not correct. Consider a hash function that provides a 128-bit hash value. There are, obviously, 2128 possible hash values. But there are a lot more than 2128 possible files. Therefore, there have to be multiple files; in fact, there have to be an infinite number of files that can have the same 128-bit hash value. The difficulty is finding two files with the same hash. What is, indeed, very hard to do is to try to create a file that has a given hash value so as to force a hash value collision which is the reason that hash functions are used extensively for information security and computer

forensics applications. Alas, researchers in 2004 found that practical collision attacks could be launched on MD5, SHA-1, and other hash algorithms. At this time, there is no obvious successor to MD5 and SHA-1 that could be put into use quickly; there are so many products using these hash functions that it could take many years to flush out all use of 128- and 160-bit hashes. Certain extensions of hash functions are used for a variety of information security and digital forensics applications, such as:

- *Hash libraries* are sets of hash values corresponding to known files. A hash library of known good files, for example, might be a set of files known to be a part of an operating system, while a hash library of known bad files might be of a set of known child pornographic images.
- *Rolling hashes* refer to a set of hash values that are computed based upon a fixed-length "sliding window" through the input. As an example, a hash value might be computed on bytes 1-10 of a file, then on bytes 2-11, 3-12, 4-13, etc.
- *Fuzzy hashes* are an area of intense research and represent hash values that represent two inputs that are similar. Fuzzy hashes are used to detect documents, images, or other files that are close to each other with respect to content.

## 5. CONCLUSIONS

Cryptography has been used in one form or another for over 4,000 years and the attacks on cryptography have probably been in process for 3,999 years and 364 days. As one group of people works to find new ways to hide and transmit secrets, another group of people is right on their heals finding holes in the newly developed ideas and products. This can be viewed as evil and destructive behavior, or hackers can be viewed as the thorn in the side of the computing world that requires them to build better and more secure products and environments. Cryptographic algorithms provide the underlining tools to most security protocols used in today's infrastructures. The algorithms work off of different mathematical functions and provide different types of functionality and different levels of security. A big leap was made when encryption went from symmetric key use to public key cryptography. This evolution provided users and maintainers much more freedom and flexibility when it came to communicating with a variety of different types of users all over the world.

**REFERENCES**

[1] A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *The Design and Analysis of Computer Algorithms*. Addison-Wesley.

[2] G. Julius Caesar, John F. Kennedy, *Security Engineering: A Guide to Building Dependable Distributed Systems*.

[3] Adi Shamir. *Identity-based cryptosystems and signature schemes. In CRYPTO*, pages 47–53.

[4] Adam O'Neill. *Definitional issues in functional encryption. Cryptology ePrint Archive*, Report 2010/556, 2010. http://eprint.iacr.org/2010/556.

[5] Shweta Agrawal, Dan Boneh, and Xavier Boyen. *Efficient lattice (h)ibe in the standard model. In EUROCRYPT*

[6] www.trincoll.edu/depts/cpsc/cryptography/index.html

[7] http://dmoz.org/Science/Math/Applications/Communication_The.ory/ Cryptography/Historical