R

Rabin Cryptosystem

David Pointcheval
Computer Science Department, Ecole normale supérieure, Paris, France

Related Concepts

►Integer Factoring; ►RSA Factoring Challenge

Definition

Rabin's public-key encryption is an asymmetric encryption scheme based on the modular square root problem, an thus related to integer factoring.

Background

The smaller the public exponent in the ►RSA public-key encryption or ►RSA digital signature schemes, the more efficient the encryption process is. Michael O. Rabin thus suggested to use $e = 2$ into an encryption scheme [3]. But things are not as simple as for RSA.

Theory

Thanks to the Euler's theorem, one can easily extract modular eth roots, until e is co-prime to $\varphi(n)$ (►Euler's Totient function) and the latter value is known: $d = e^{-1} \bmod \varphi(n)$ helps to get it. Unfortunately, $e = 2$ is not co-prime to $\varphi(n)$, moreover squaring is not a bijection in the ►group $\mathbb{Z}_n^*$, for $n = pq$ (►modular arithmetic), and even in $\mathbb{Z}_p^*$ for a ►prime number p: if x is a square root of y in $\mathbb{Z}_p^*$, then $-x$ is also a square root of y. More formally, the function $f : x \mapsto x^2 \bmod p$ from $\mathbb{Z}_p^*$ into $\mathbb{Z}_p^*$ is a morphism, whose kernel is $\{-1, +1\}$. As a consequence, the cardinality of the image of f is exactly $(p-1)/2$: An element in $\mathbb{Z}_p^*$ is either a square with two square roots, or a non-square without any square root (►quadratic residue).

Once again, the ►Chinese Remainder Theorem helps to know more about squares in $\mathbb{Z}_n^*$ for a composite $n = pq$. Indeed, y is a square in $\mathbb{Z}_n^*$ if and only if it is a square in both $\mathbb{Z}_p^*$ and $\mathbb{Z}_q^*$: there are only $\varphi(n)/4$ squares, which admit four distinct square roots: $x, -x, z$ and $-z$. Since they are distinct, $x + z \neq 0 \bmod n$ and $x - z \neq 0 \bmod n$. However, $x^2 = z^2 = y \bmod n$. Then,

$$x^2 - z^2 = (x - z)(x + z) = 0 \bmod n.$$

As a consequence, $\gcd(x - z, n) \in \{p, q\}$: The ability to compute modular square roots helps to factor the modulus.

In the other direction, Euler's theorem does not help any more, since 2 is not co-prime to $\varphi(n)$ (whatever n is, either a prime or a composite integer). But first, for prime moduli, methods are known to compute square roots. Particularly, for *Blum primes* p, which satisfy $p = 3 \bmod 4$ (►Blum integer), if y is a square in $\mathbb{Z}_p^*$, then the square roots are $\pm y^{(p+1)/4} \bmod p$. Then, for computing square roots in $\mathbb{Z}_n^*$, one can simply use the Chinese Remainder Theorem: from $y \in \mathbb{Z}_n^*$, one uses the isomorphism from $\mathbb{Z}_n^*$ onto $\mathbb{Z}_p^* \times \mathbb{Z}_q^*$. One then computes the square roots in $\mathbb{Z}_p^*$ and $\mathbb{Z}_q^*$. The inverse isomorphism on the four possible pairs leads to the four possible square roots of y. Therefore, the square root problem in $\mathbb{Z}_n^*$, with $n = pq$, is equivalent to the factorization of n, which is a stronger formal result than for RSA.

The Rabin Primitive

Granted the equivalence between the modular square root problem and the factorization of the modulus, it is natural to try to use it for cryptographic applications: Rabin suggested a public-key cryptosystem [3].

- Key generation: Randomly choose two large Blum primes p and q, and compute $n = pq$. The public key is thus the modulus n, while the private consists of its factorization (p, q).
- Encryption: In order to encrypt a message $m \in \mathbb{Z}_n^*$, one computes $c = m^2 \bmod n$.
- Decryption: Given the ciphertext c, granted the factorization of n, one can extract the square roots.

Unfortunately, a problem arises here because of the non-injectivity of the square function: Four plaintexts are possible. Redundancy in the plaintext is thus required to help the recipient to make a choice. Furthermore, the algebraic structure allows several kinds of attacks, as RSA suffers [5], and thus paddings are required to solve the two problems.

The SAEP$^+$ padding [2] that contains redundancy can be applied to the Rabin primitive. It would then lead to an efficient encryption scheme, provably IND–CCA2 secure [4] under the intractabilty of integer factoring, in the ►random oracle model [1].

Recommended Reading

1. Bellare M, Rogaway P (1993) Random oracles are practical: a paradigm for designing efficient protocols. Proceedings of the 1st CCS. ACM Press, New York, 1993, pp 62–73
2. Boneh D (2001) Simplified OAEP for the RSA and rabin functions. In: Kilian J (ed) Advances in cryptology–CRYPTO 2001. Lecture notes in computer science, vol 2139. Springer, Berlin, pp 275–291
3. Rabin MO (1978) Digitalized signatures. In: Lipton R, De Millo R (eds) Foundations of secure computation. Academic, New York, pp 155–166
4. Rackoff C, Simon DR (1992) Non-interactive 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, Berlin, pp 433–444
5. Rivest R, Shamir A, Adleman L (1978) A method for obtaining digital signatures and public key cryptosystems. Commun ACM 21(2):120–126

Rabin Digital Signature Scheme

Dan Boneh
Department of Computer Science, Stanford University, Stanford, CA, USA

Related Concepts

►Digital Signature Scheme; ►ElGamal Digital Signature Scheme; ►Schnorr Digital Signature

Background

In [5] Rabin described a ►trapdoor one-way function that can be used for ►digital signatures and for public-key encryption (►public-key cryptography). Here we focus on Rabin's digital signature system.

Theory

Rabin's trapdoor function makes use of ►modular arithmetic and is defined as follows: (1) let $N = pq$ be a product of two distinct equal size ►prime numbers, (2) define the function as $F : Z_N^* \to Z_N^*$ as $F(x) = x^2 \in Z_N^*$. Since N is a product of two distinct primes, the function F is a 4-to-1 map on Z_N^* (every element in the image of F has exactly four pre-images). Rabin shows that inverting this function is as hard as factoring the modulus N. However, given the factorization of N it is easy to find all four pre-images for a given element in the image of F. Hence, the factorization of N serves as a trapdoor for this function.

In the ►random oracle model, there are several generic methods for building a secure signature scheme from a 4-to-1 trapdoor function. Here we describe Rabin signatures using the Full Domain Hash method [2].

Key Generation. Given a security parameter $\tau \in Z$ as input do the following:

1. Generate two random τ-bit primes p, q where $p = q = 3$ mod 4. Set $N = pq$.
2. Pick an element $w \in Z_N^* t$ such that the ►Jacobi symbol of w over N is equal to -1. In other words, w is a ►quadratic residue modulo exactly one of p or q.
3. Let H be a ►hash function $H : \{0,1\}^* \to Z_N$.
4. Output the public key (N, w, H) and the private key (N, p, q, w, H).

Signing. To sign a message $m \in \{0, 1\}^*$ using the private key (N, p, q, w, H) do:

1. Compute $x = H(m) \in Z_N$. If x is not in Z_N^* output "fail" and abort. This is extremely unlikely to happen.
2. One can show that exactly one of $\pm x, \pm xw \in Z_N^*$ must be a quadratic residue. Let $y \in \{\pm x, \pm xw\}$ be that value. To find y, find the unique element in $\{\pm x, \pm xw\}$ for which the ►Legendre symbol is equal to 1 over both p and q.
3. Let $s \in Z_N^*$ be the square root of y in Z_N^*. Output s as the signature on m.

Verifying. To verify a message/signature pair $(m, s) \in \{0,1\}^* \times Z_N$ using the public key (N, p, q, w, H) do:

1. Compute $x = H(m) \in Z_N$.
2. Check if $s^2 \in \{\pm x, \pm xw\}$. If so, accept the signature. Otherwise, reject.

Note that signature verification is fast requiring a single modular squaring. When the hash function $H : \{0,1\}^* \to Z_N$ is modeled as a random oracle, one can show that the signature scheme is existentially unforgeable under a chosen message attack assuming that factoring random ►Blum integers is intractable [2, 4]. To do so, one shows that a forging algorithm can be used to factor the modulus N in the public key.

We note that Rabin signatures can be shortened by a factor of 2 using a cute trick due to Bleichenbacher [1]. The basic idea is to output only half the bits of s (the most significant ones). Let $\hat{s}$ be the resulting signature. Its length is τ-bits as opposed to 2τ-bits. During verification, the least significant bits of the signature can be recovered using Coppermith's algorithm [3]. Indeed, given $x, \hat{s} \in \mathbb{Z}$, Coppersmith's algorithm can test whether there exists a $0 \leq \Delta < 2^{\tau}$ such that $(\hat{s}\,2^{\tau} + \Delta)^2 = x \bmod N$. If Δ exists the algorithm will find it, thus recovering the missing bits of the signature. For a more efficient method, see [1].

Recommended Reading

1. Bleichenbacher D (2004) Compressing Rabin signatures. In: Okamoto T (ed) Proceedings CT-RSA, Topics in cryptology: CT-RSA 2004. Lecture notes in computer science, vol 2964. Springer-Verlag, Berlin, pp 126–128
2. Bellare M, Rogaway P (1996) The exact security of digital signatures: how to sign with RSA and Rabin. In: Maurer U (ed) Advances in cryptology – EUROCRYPT'96. Lecture notes in computer science, vol 1070. Springer-Verlag, Berlin, pp 399–416
3. Coppersmith D (1997) Small solutions to polynomial equations, and low exponent RSA vulnerabilities. J Cryptol 10:233–260
4. Jean-Sébastien C (2000) On the exact security of full domain hash. In: Bellare M (ed) Advances in Cryptology – CRYPTO 2000, Lecture notes in computer science, vol 1880. Springer-Verlag, Berlin, pp 229–235
5. Rabin M (1979) Digital signatures and public key functions as intractable as factorization. Technical Report MIT/LCS/TR-212, MIT Laboratory for Computer Science, Cambridge, MA

Radio Fingerprinting

►Wireless Device Fingerprinting

Radio Frequency Attacks

Jean-Jacques Quisquater[1], David Samyde[2]
[1]Microelectronics Laboratory, Université catholique de Louvain, Louvain-la-Neuve, Belgium
[2]Intel Corporation, Santa Clara, CA, USA

Definition

All attacks on the functioning of an electric circuit by means of electromagnetic field.

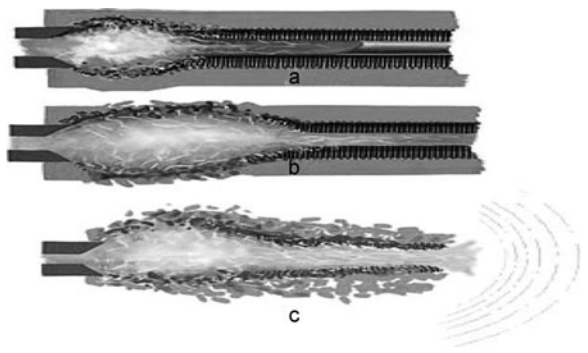
Background

Summary: Conquest without fight is the basic idea of Sun Tzu in *The art of war* written 2,500 years ago. Force an enemy to retreat immediately, strike with high precision without leaving of the origin of the blow, shut down an adversary's communications networks, disrupt its power supplies, yet still leaving buildings intact are the properties conferred to an attack that uses electromagnetic waves at the right frequencies. Most types of matter are transparent to microwaves, and waves coming from an electromagnetic blast are difficult to stop in an appropriate manner. A mastered generation of microwaves may not only disrupt or damage electronic equipment, but may also create faults and even completely destroy it. Solar storms constitute a good illustration of the kind of disruption that an electronic equipment might be submitted to when exposed to electromagnetic disruptions (satellite communications, etc.).

Today's computers and other electronic devices are sensitive to computer attacks such as worms, viruses, or logical bombs (►Trojan Horses, Computer Viruses and Worms). Electromagnetic radiation leakage is well known and remains the subject of studies for some devices [1, 7, 12, 15]. The electronic components that make up common devices may, however, be disrupted by intense electromagnetic variations in their near surroundings. The sensitiveness of the equipment has thus to be taken into account. Electromagnetic waves that are able to destroy electronics (Compton effect) can be obtained by different means. They may be used for specific purposes by determined will (e-bomb Mk-84, Argus Project on the 27th of August 1958), but they may also have accidental results (Blackout of Hawaii and disturbances in radio-navigations in 1958). The imagination of scenarists and the movie world made the general public become aware, not only of the existence but also of the power, of Electromagnetic Pulse (EMP), through Hollywood films like Golden Eye (1995), Broken Arrow (1996), Matrix (1999) or Ocean's Eleven (2001). This type of attack is, however, not yet very common and seems to be part of advanced research projects [5]. This may be due to the particular constraints they involve like mastering impulse modes, high currents, and high temperature supraconductivity. Anyway, the technology will however become more available and the number of incidents will increase. Has a new Pandora's box been opened [8]?

Theory

Introduction and Definitions: A sufficiently intense and correctly directed electromagnetic field may disrupt the functioning of an electric circuit to different degrees. The circuit may become completely useless. According to Carlo Kopp [10], electromagnetic weapons can nowadays be realized by nations with limited powers, close to those at the disposal of the big nations at the beginning of World War II. More than 20 countries have development programs for



Radio Frequency Attacks. Fig. 1 Courtesy of John Batchelor © http://www.jbatchelor.com

some kind of radiofrequency weapons. These weapons do not leave any trace behind, damage buildings, may strike without anybody noticing it a priori, and may be created by ill-intentioned people without yielding any big benefit, but generating enough power to cause delicate situations. The idea of using fields for destruction purposes is rather old and today it has become possible for an amateur to construct an e-bomb in his garage. The Internet also contributes to the dissemination of technical data related to this technology [6, 7], and details the materials and open literature or reference material to a high number of persons.

There are two types of RF attacks, namely, high and low power attacks. The high power attacks are the most devastating ones, requiring mastering techniques like explosively pumped flux generator (Field Compress Generator), homopolar generators, plastic explosives, high performance detonation, and magnetohydrodynamics (MHD). This is why they are almost inaccessible to nongovernmental entities, but almost exclusively reserved to government-funded, specialized research units (or to government supported terrorists). Moreover, their final bulk and form factor are rather imposing and close to the size of more conventional bombs. Attacks using low power weapons, on the contrary, are considerably easier, less expensive (really very low cost) and are accessible to many people, provided that they dispose of the required level of expertise. An experimented technician or an engineer can design, fabricate, and experiment such a device. Very old antennas and military amplifiers can, for instance, be bought from military surplus and can easily be modified to serve these purposes. It is not even necessary to look for components that have this high potential, capable of yielding this kind of power.

Ultrawideband and narrowband weapons are the basis of radiofrequency attacks. Ultrawideband devices emit over a large frequency range; nanoseconds long burst of low energy (about 10 J/pulse) are radiated hundred of times within a second. Their destructive power is directly linked to the strength of the source and to the distance from the target. On the contrary, narrowband weapons emit a very reduced spectrum or a unique sinus wave (one frequency) at very high power which can reach thousand kilo-joules per pulse and hundreds of times a second.

Electromagnetic waves, like acoustic waves, may however interact with human and other living beings. Radiotherapy treatment of cancers is a proof thereof, without even mentioning food cooking in a microwave oven. In the same way as audio waves with frequencies of a few Hertz, microwaves may have a disturbing effect on the human organism (VMADS and crowd control). But this is not the subject of this article. It is important not to confuse instruments allowing conducting an RF attack with jamming devices that are used to temporarily disrupt the electromagnetic spectrum within some frequency span and in a given region. This is not the subject of this article either.

Principles and Description

VLSI stands for Very Large Scale Integration, and the integration density of modern chips is ever growing (system on chip, etc.). Modern VLSI chips are extremely sensitive to voltage surges, and could be burned out by even small current leakages resulting from an EMP. The higher the circuit's density, the more vulnerable it is of course. It is also important to notice that, as the power voltage of chips and computers is going down and down, their susceptibility is increasing.

ElectroMagnetic Pulse

Arthur Compton discovered the Compton effect in 1923. When a beam of X-rays of well-defined frequency is scattered through an angle by sending the radiation through a metallic foil, the frequency of the scattered radiation is different from the original one. So if a nuclear blast occurs at high altitudes, the gamma rays following the detonation are the source of HEMP. When the rays encounter the upper regions of the atmosphere, the molecules interact with the rays depending on the atmospheric density and burst conditions. So the energy of the gamma rays is transferred to the electron of an air molecule. The EMP effect may, in fact, decompose into three periods: early time (0–1 μs), intermediate time (1 μs to a tenth of a second), and, last but not the least, late time, which involves magnetohydrodynamic properties. At the first order, gamma radiations cause bursts of electrons from the photoelectric

Compton effect. Contrary to high altitude detonation, a surface burst EMP (SBEMP) can be produced by a nuclear burst close to the earth's surface. In such a case, particular attention must be given to surge protection to dissipate the high currents. Ground connection can enable in certain cases to reduce or send back the currents that are created by the EMP. An air-burst EMP is the third possibility between low and high altitudes (0.5–30 km). The effects are a melt of both previous cases.

System-generated EMP (SGEMP) and Internal EMP (IEMP) are of course obtained from the direct interaction of nuclear gamma rays and X-rays from a system. The first case is very important outside the atmosphere, typically for satellites. The emission of electrons from internal EMP can create current generation and electromagnetic fields within cavities create the second one.

Coupling

Energy distribution lines or telecommunication wires are omnipresent, both at the level of buildings and within electrical circuits. This is why they are privileged targets for radio frequency attacks. As the emitted wave rapidly attenuates, it is more interesting to create a surge in a wire and let it propagate, rather than to increase indefinitely the emission power (which implies very complex problems). But therefore one should be able to optimize the coupling between the radiated wave and the circuit that will receive it. Water supply circuits or metal pieces in a building, without electrical function, may also help EMP-induced currents to transit (dangerous in case of cross talk).

It is sometimes necessary to call upon theoretical physics and electronics to compute the equivalent circuit to determine the EMP-induced voltage. Electronic structures are often approximated and it is not always possible to consider that the radiation is a plane wave. So after the experiment, the empirical data should be compared with the experimental data coming from complex analytical computations. Transient effects also have to be taken into account and therefore typical coupling models and shielded cables coupling models are used. These models consider the transfer impedance and the conductivity, when they can be calculated.

Transmission line theory is nevertheless the most commonly used method to determine the effects of an EMP on aerial and buried conductors. The length of the conductor, then, has to be compared with the length of the radiated wave, which is one of the principal characteristics together with the characteristic impedance and the charge impedance. Computers, then, allow finding an analytical solution.

All electronic circuits contain resistors, capacitors, or inductances, which provide them with one or several resonance frequencies. In this way, the electromagnetic pulse can make the circuit resonating at several dominant frequencies. According to the electrical resonance, the oscillation can be long lived, but this highly depends on the shielding and grounding characteristics.

It is also important to notice the cross talk effects. A cable that conveys current can, depending on its distance to other surrounding cables (water pipe, twisted pair, ...), create by radiation a current in another cable that was initially not affected by the problem.

Damage

The equipment's susceptibility is an important parameter to know if it can be avoided, affected, disrupted, or destroyed by a radio frequency attack. Because of Ohm's law, high impedance and low voltage signals are most susceptible to interfere. The obtained currents are sometimes even sufficient to fuse the silicon on the chip. We can, however, distinguish several entering points for a device under test. The first one is called the front door and concerns elements for which the nature or structure favors the effects of the waves. Computer or other electronic devices have attached antennas that constitute a nice front door for an electromagnetic pulse to penetrate into a device [13]. The back door, on the contrary, may be composed of an unshielded wire.

Damages may thus be of different natures, going from disruptions to lasting destruction, including thermal related failure, metallization burnout, and the avalanche effects in active components [9, 19]. Passive components are destroyed by voltage breakdown or induced thermal overstress. This is true for resistors (overheating) and capacitors (dielectric breakdown) [18].

Even for cases where the result would not be directly exploitable, the real problem in the use of an RF attack by nonexperts would be the Denial of Services (DOS). Without reaching the needed destruction level, it could be possible to damage an unprotected and important structure. The replacement or the repairing time may then be prohibitive with respect to the function of the machine.

Energy

Maxwell's law explains that the effect of an electromagnetic wave follows an inverse square law with increasing distance, so the strength of the wave dissipates quickly as it moves away from the initial point of the blast. It is thus necessary to get substantial energies (GigaWatts) in very short times to create electromagnetic disturbances such as the ones described in this entry. The sources of the energies put at stake may be twofold: nuclear or electrical energy, but even sometimes converted mechanical energy. Obviously, the results are better when using nuclear energy.

Ultrawideband bombs create en electromagnetic pulse like a nuclear detonation, but a conventional or chemical explosive replaces the nuclear part. The microwave source relies on an extremely fast switching device. But narrow band bombs are based on magnetron or vircator (Virtual Cathode Oscillator).

Nonnuclear Techniques

Forty years ago, Andréï Sakharov (Nobel Peace Price 1975) has elaborated, together with Altshuler, Voitenko, and Bichenkov, and at the same time as Clarence Marx, the first explosively pumped flux generator (FCG) [14]. He obtained an intense field by discharging a capacitor inside a solenoid and managed to crush the field lines with a peripheral explosive. Then, also by using the energy of an explosive, he constructed the MK-2. The locking up of the explosive within a copper tube containing an inductance, made it to deform the tube by short-circuiting the spirals of the solenoid one after another. As the speed is very high, the sudden decrease of the inductance and the conservation of the flux occasion a brutal increase of the intensity. Very big FCGs have already reached some tens of gigawatts, and as for magnetrons, they can be cascaded, the output of the first one supplying the entry of the second.

The microwave source used is an extremely fast-switching device. Narrowband e-bombs use a virtual cathode oscillator tube or a variant of a magnetron. The idea behind the Vircator is accelerating a high current electron beam against a mesh anode. Many electrons will pass through the anode, forming a bubble of space charge behind the mesh anode. Under the proper conditions, this space charge region will oscillate at very high frequencies. If the space charge region is placed into a resonant cavity which is appropriately tuned, very high peak powers may be obtained.

Nuclear Techniques and Transient Electromagnetic Devices (TED): Schamiloglu's team and the US Air Force's Shiva Star is a pulsed-power system used to simulate the effects of nuclear weapons. TED are very simple devices at low cost. TED does not generate a pure sine wave, it operates completely differently than narrow band devices. Instead of generating a burst of smooth sinus it generates a single spike of energy.

Protection and Countermeasures

The first countermeasure consists in performing a real measure of the susceptibility of the involved equipment. Constructing an analytical model of the disruptions that can be generated on a device is very difficult; this is the reason why tests are carried out. The Federal Communication Commission (FCC) establishes the testing method for emission certification of commercial products. A lot of FCC measures are indeed carried out nowadays and in the best cases without adding all possible options to the equipment. For commercial purposes, the great majority of present machines are constructed in such a way as to respect the electromagnetic compatibility standards. Each year, the aircraft direction notices, however, cases where portable computers or electronic gadgets manage to disrupt the aircraft's electronics. The same applies to machines that are used in hospitals and their interaction with portable telephones. So, the required level of protection is to be evaluated before starting to devise and apply a whole series of countermeasures.

Computers are very sensitive to RF attacks, because these can be propagated through their power cable, network interface, the wire mouse, the keyboard, input/output slots and cables, and all kinds of apertures (peripherals, buttons, ventilation, etc.). Telecommunications Electronics Material Protected from Emanating Spurious Transmissions (TEMPEST) are better protected than ordinary computers, because the reduction of their radiation level also reduces their sensitiveness. Machines abiding by the military "Milspec" norms, or "ruggedised" machines, already exist, but their power and cost discourages possible buyers. Solutions exist nonetheless, but they often involve high costs. The use of optical fibers would, for instance, highly reduce the sensitiveness of machines. Nowadays, nothing forbids the use of a keyboard or a mouse that uses optical communication from one side to the other, like micro-spies already do by modulating the light signal of an optical fiber. The construction of a comprehensive shield and the use of ferrite beads, together with a nonelectrical coupling of the connectors would already reduce the risks. Another advantage would be the reduction of mass loops, which often disrupt the connection of peripherals. A commercial computer, armored against RF attacks, does not need the same level of protection as a military machine, but it requires to continue working in the presence of disruptions on, for example, the power supply lines.

PC switch-mode supplies are particularly sensitive to electromagnetic disruptions, because they all contain components that are ideally sensitive to an attack. It is, however, possible to lower their fragility by modifying slightly the structures of their power supplies. A perfect power supply would recreate its energy in a Faraday cage. One should therefore use a dynamo that is mechanically coupled to a motor, situated outside the room.

Network components are also hardly protected, and cables, scattered around entire buildings, allow conveying the induced signal. Commutation electronics that are used in network devices are, of course, very sensitive to RF attacks. The use of optical fibers, which convey light but do not offer electrical conduction, would allow reducing the risks. Optical fibers are immune against any kind of electromagnetic attack.

These attacks disrupt or damage logic devices, but there are some components that they are not in power to act upon. Vacuum tubes were used during the conquest of space for their weak sensitiveness to solar radiation disruptions. These tubes do not use silicon as a basic component, so the Compton effect does not create any noticeable disruption on their functioning. It should be noticed that some fight aircraft have still some of their sensitive parts equipped with tubes in order to avoid electromagnetic disruptions. These tubes suffer, however, from some other drawbacks, such as their mechanical resistance.

Perfect protection against RF attacks consists in a complete isolation of the sources of disruption. Therefore, one should use a Faraday cage. The Faraday cage allows protecting sensitive equipment by using a metallic and ferromagnetic box that closes hermetically. External disruptions cannot enter the cage. A good compromise to perform an opening is to make a hole with sufficiently small diameter into the Faraday cage and adding a tube with a length of eight times the diameter of the hole. Otherwise, the classical RF traps or the ferrites grommets should be used. If the inside of the cage is to be ventilated, the air arrival should be extremely well controlled in order to prevent undesirable radiations from entering. In the same way, the communication with the outside should be performed through optical fibers. The envelope of Faraday cages is in general connected to the ground, except for those that are situated at some height because of the ground effect that they would engender. The risks involved in the cold war have made these techniques to be already mastered since a long time.

The price of a Faraday cage is very high and it is not always easy to put them in place. So the protection level has sometimes to be degraded, contenting oneself with electrostatic shielding. Minimal protection consists in rounding the cables with a flexible metal sheath. Simply connecting these loops to the ground is not enough. The electromagnetic compatibility has to be taken into account (but this is not the subject here).

Targets and Properties

Potential targets of RF attacks are manifold. Financial systems, telecommunications, medical centers, critical infrastructures as airport radars, transportation means (aircrafts, electronic ignition cars) are all aimed at.

Properties

►Radio frequency weapons have, compared to their drawbacks (Hugh power consumption), numerous advantages. First of all, these weapons are tunable (which is useful when a target seems to be invulnerable to particular frequencies) and some are even reusable. But they can be triggered off and fired from miles away at any time and in every circumstance. Contrary to weapons that need ballistic computations, an RF attack is gravity independent and hardly detectable. These are low-cost weapons and the materials needed could easily be acquired in a large city, without conventional counterterrorist agencies being able to trace them easily. But above all, these weapons allow multiple target acquisition and do not concentrate on a single target, they are instantaneous and non-lethal to humans, which makes them very adapted to denial of service attacks.

Further Reading

We will present here our work based on the perturbation of a smart card or a crypto processor using intense electromagnetic fields, our aim being to create a fault in order to apply cryptanalytic methods [4] such as Differential Fault Analysis [3]. Our method also permits destroying a chip, but there are very few advantages in doing so. Attackers generally want to create transient faults in a cryptosystem, and permanent faults are quite rare (security sensor destruction, etc.).

Traditional electromagnetic analysis is based on a sensing coil located in the near field of the chip. Measurement is thus passive. In some circumstances, this gives similar information to that obtained by measuring the chip's power consumption (i.e., in power analysis). It has, however, been shown that electromagnetic analysis gives strictly more information, as the coil can pick up the magnetic fields generated by local signals that are not present outside the chip. This was highly significant in itself; it also turned out to be important for later analysis and protection work. An alternating current in a coil near a conducting surface creates an electromagnetic field. Here we send a high current in a coil very close to the chip, so it is an active measurement. An active measurement can interfere with the local or total activity of the processor. Depackaging the chip is not necessary to apply this attack, but the ability to see the surface of the chip improves the precision. The application of the traditional techniques of depackaging with concentrated nitric acid and acetone is still an easy way to open many chips [Anderson].

We developed a variant of this technique using their electromagnetic probing tools. By placing a small coil next to a target component in a smartcard chip and passing a current pulse through the coil, they found that they could induce a sufficiently large eddy current in the chip to cause a targeted malfunction.

Our sensor is composed of a touch point coming from a microscope and a wire. The wire is wound on the test probe. The current injected in the coil creates an electromagnetic field. The test probe concentrates the lines of the field. So the field obtained at the end of the needle is relatively intense. The current injected into the coil can be obtained by using a simple camera flash gun. The intense magnetic field allows moving charges, so we create a movement of charges through the grid oxide of the transistors. Charges are then stocked in the grid oxide by a tunnel effect with high energy.

At present, silicon manufacturers shrink transistors to increase their density on a chip. But if the reduction in the thickness of the grid oxide is too high, the transistor will cease functioning. Thus we imagine that our attack will be easier to use in the future although its precision will be reduced. In order to quantify the number of charges brought to or withdrawn from the grid oxide, we decided to heat the components. The increase in the electronic shocks resulting from the increase in temperature could make the quantity of loads present in the grid oxide evolve. For a static RAM, commercially available from a big silicon manufacturer, 95.81% of the remaining faults were maintained after 100 h at 420 °K. It seems not trivial to stop this attack, and designers should continue to use hardware and software countermeasures.

Conclusions

Electromagnetic attacks find their theoretical foundations in the basic physics introduced by Faraday and in the work of Compton, Einstein, Oppenheimer, Sakharov, Marx, and many others that have contributed to finalizing these weapons. The world's principal armies already have these devices at their disposal and others are trying to obtain them. The security of existing systems is not always conceived in a way to resist this type of attacks. By highly reducing the order of magnitude, it is also possible to use the basic principles of these attacks for inserting faults or disrupting the functioning of a cryptographic machine. Other attacks, for instance, glitch insertion on power supplies, clock disruption (metastability) or optical attacks [17] are however also possible.

Intense electromagnetic fields can easily be created and may allow disrupting the good functioning of electronic components. The required components are commonly used and can be easily obtained. It is possible to disrupt a cryptographic component by acting close to the device or to disrupt an entire building at a distance of some hundred meters, which of course depends on the level of energy. Using fields leaves behind few traces, and finding back their source is a very difficult task. Unfortunately, many devices offer a nonzero coupling and are thus a very easy victim for electromagnetic disruptions. The damage can then be very important and may require the replacement of the electronics. Countermeasures exist, but are not often used.

The security domain has to take into account the possibilities of pirating by RF disruptions. The attacks can spread from a simple deny of service to the destruction of a machine, passing by disruptions that are more difficult to detect. The interaction with the cryptographic world is without any doubt to be situated in the very local use of intense fields, used to influence the functioning of cryptoprocessors or ciphering machines.

Recommended Reading

1. Agrawal D, Archambeault B, Rao JR, Rohatgi P (2002) The EM side-channel(s). In: Kaliski BS Jr, Koç CK, Paar C (eds) Proceeding of the cryptographic hardware and embedded systems CHES 2002, Redwood City. Lecture notes in computer science, vol 2523. Springer, Berlin, pp 29–45. Also available at http://ece.gmu.edu/crypto/ches02/talks.htm
2. Anderson R, Kuhn M (1996) Tamper resistance—a cautionary note. In: Proceedings of the second Usenix workshop on electronic commerce, pp 1–11. Also available at http://www.cl.cam.ac.uk/~mgk25/tamper.pdf
3. Biham E, Shamir A (1997) Differential fault analysis of secret key cryptosystems. In: Kaliski B (ed) Advances in cryptology – CRYPTO'97, Santa Barbara. Lecture notes in computer science, vol 1294. Springer, pp 513–525. Also available at http://citeseer.nj.nec.com/biham97differential.html
4. Boneh D, Demillo RA, Lipton RJ (1997) On the importance of checking cryptographic protocols for faults. In: Fumy W (ed) Advances in cryptology – EUROCRYPT'97, Santa Barbara. Lecture notes in computer science, vol 1233. Springer, pp 37–51. Also available at http://citeseer.nj.nec.com/boneh97importance.html
5. Bresselin S (2003) Une avancée franco-allemande vers la bombe "E". Air & Cosmos, No. 1877, 14 February 2003
6. http://www.cryptome.org
7. Gandolfi K, Mourtel C, Olivier F (2001) Electromagnetic attacks: concrete results. In: Koç ÇK, Naccache D, Paar C (eds) Proceedings of the cryptographic hardware and embedded systems CHES 2001, Paris. Lecture notes in computer science, vol 2162. Springer, pp 251–261. Also available at http://www.gemplus.com/smart/r_d/publications/pdf/GMO01ema.pdf
8. Abrams M (2003) Dawn of the E-Bomb. IEEE Spectrum 2003. Also available at http://www.spectrum.ieee.org/WEBONLY/publicfeature/nov03/1103ebom.html
9. Jenkins CR, Durgin DL (1975) EMP susceptibility of integrated circuits. IEEE Trans Nucl Sci NS-22(6)
10. Kopp C (1997) Hardening your computing assets. Asia/Pacific Open Systems Review. Computer Magazine Group, NSW under

the title of "Information Warfare—Part 2," Australia. Also available at http://www.globalsecurity.org/military/library/report/1997/harden.pdf

11. Kopp C (1996) The electromagnetic bomb—a weapon of electrical mass destruction. USAF CADRE Air Chronicles, Melbourne. Also available at http://www.airpower.maxwell.af.mil/airchronicles/kopp/apjemp.html
12. Kuhn MG, Anderson RJ (1998) Soft tempest: Hidden data transmission using electromagnetic emanations. In: Proceedings of information hiding, second international workshop, IH'98, Portland, pp 124–142. Also available at http://www.cl.cam.ac.uk/~mgk25/ih98-tempest.pdf
13. Lee KSH, Liu TK, Morins L (1978) EMP Response of aircraft antennas. IEEE Trans Antenna Prop IEEE AP-26, 1
14. Petit J-P (2003) Armes secrètes américaines – L'extraordinaire témoignage d'un scientifique. Collection Aux marches de la science, Albin Michel, ISBN: 2226136169
15. Quisquater J-J, Samyde D (2001) Electro-Magnetic Analysis (EMA): Measures and countermeasures for smart cards. In: Attali I, Jensen T (eds) Proceedings of the international conference on research in smart cards E-Smart, Cannes. Lecture notes in computer science, vol 2140. Springer, pp 200–210
16. http://www.voltsamps.com
17. Skorobogatov S, Anderson R (2002) Optical fault induction attacks. In: Kaliski BS Jr, Koç CK, Paar C (eds) Proceedings of the cryptographic hardware and embedded systems CHES 2002, Redwood City. Lecture notes in computer science, vol 2523. Springer, pp 2–12. Also available at http://ece.gmu.edu/crypto/ches02/talks.htm
18. Tasca DM, Wunsch DC, Domingos H (1975) Device degradation by high amplitude currents and response characteristics of discrete resistors. IEEE Trans Nucl Sci NS-22 (6)
19. Wunsch DC, Bell RR (1968) Determination of threshold failure levels of semiconductor diodes and transistors due to pulse voltages. IEEE Trans Nucl Sci NS-15(6)

Radio Interference Attack Defense

►Jamming Attack Defense

Radio-Frequency (RF) Fingerprinting

►Wireless Device Fingerprinting

Radiometric Identification

►Wireless Device Fingerprinting

Radiometrics

►Wireless Device Fingerprinting

Rainbow Tables

Gerald Brose
Hype Softwaretechnik GmbH, Bonn, Germany

Related Concepts

►Password

Definition

A rainbow table [1] is a precomputed lookup table used to retrieve plaintext passwords in an attack. Rainbow tables offer a space-time trade-off and are used as an attack preparation when either storing a complete lookup table of all possible password values would consume too much storage, and an exhaustive search is computationally infeasible.

Theory

Precomputing all possible password hash values for an exhaustive search and storing them in a lookup table drastically reduces the time required for an attack. This approach is usually not possible for large search spaces, where the resulting tables would exceed the available storage space. The time-memory trade-off consists of combining partially stored tables with computing the relevant missing parts at attack time. Rainbow tables are actually an optimization of an earlier technique described in [2].

Rainbow tables organize precomputed password hashes in hash chains of a given length. To create a hash chain, not only the cryptographic hash function H but also a reduction function R need to be defined. Note that R is not an inverse of H, as such an inverse cannot be assumed to exist for a cryptographic hash function.

A hash chain of length k is created using an arbitrary start value h and applying H to h, then R, and so on, until after k function applications a value x is found and stored as an end point together with the start point p. Given the starting point p, the chain can now be recomputed any time, thus revealing both the plaintext input to H that yields x and a number of intermediate hash values.

To retrieve the plaintext p such that H(p) yields a given hash value x, one has to find a hash chain containing x. The plaintext will then be x's *predecessor* in the chain. To determine if x is contained in any of the stored chains, it has to be tested if a *successor* of x is stored in the list of end points. This is tested by applying the functions R and H alternatingly to x. If a known endpoint is found, the complete chain is recomputed from its starting point, and x's predecessor in the chain is returned as the result, i.e., the recovered password. The required storage for this approach is much less than for a complete lookup table because only the start and end values of a hash chain need to be stored.

The main problem with this approach is that chains may merge where the reduction function has collisions, which significantly reduces the savings effect. To avoid collisions, a set of different reduction functions $R_1 \dots R_n$ is used so that for each position in the chain a different R is applied. This sequence of reduction functions can be seen to metaphorically resemble the bands of the rainbow and hence gave this technique its name.

Rainbow tables have been successfully used in attacks on Windows NT/2000 passwords, which do not use ►salts.

Recommended Reading

1. Oechslin P (2003) Making a faster cryptanalytical time-memory trade-off. In: Advances in cryptology: proceedings of CRYPTO 2003, 23rd annual international cryptology conference Lecture notes in computer science, Springer, Berlin
2. Hellman M (1980) A cryptanalytic time-memory trade-off. IEEE Trans Inform Theory 26(4):401–406

Random Bit Generator

Marco Bucci
Fondazione Ugo Bardoni, Roma, Italy

Definition

A random bit generator is a system whose output consists of fully unpredictable (i.e., statistically independent and unbiased) bits. In security applications, the unpredictability of the output implies that the generator must be also not observable and not manipulable by any attacker.

A random bit generator basically differs with respect to a ►pseudorandom number generator, because the complete knowledge of the generator structure and of whatever previously generated sequence does not result in any knowledge of any other bit. This means that the entropy of a sequence of n output bits should be ideally equal to n. On the contrary, the entropy of a sequence of n output bits of a pseudorandom generator cannot be greater than its seed, whatever n is.

Since pseudorandom generators are suitable in those applications where just a flat statistic is needed, random generators are suitable in applications where unpredictability is also needed.

Background

A true random bit generator has necessarily to be based on some kind of nondeterministic phenomena that could implement the source of the system randomness. The random sources commonly used can present several statistic defects, due to physic limitations, implementation issues, or to extern attacks aimed to manipulation. For this reason, usually, the scheme of a random bit generator consists of a raw random stream source and of a post-processor as shown in Fig. 1.

The post-processor compresses the sequence produced by the source, so that it distills the entropy (►Information Theory) in the outgoing sequence. The amount of required compression depends both on the effective entropy of the source and on the efficiency of the post-processing algorithm. Of course, since the post-processor cannot oppose every malfunction or attack attempt, the random bit generator should be provided with alarm sensors capable of revealing those anomalies for which the post-processing cannot compensate.

It is worthwhile to note that even if the post-processing uses cryptographic functions similar to those used by pseudorandom generators, it works in an opposite way. In fact, a pseudorandom generator expands its input (i.e., its seed); the post-processing, on the contrary, compresses it.

Theory

Raw Random Bit Sources

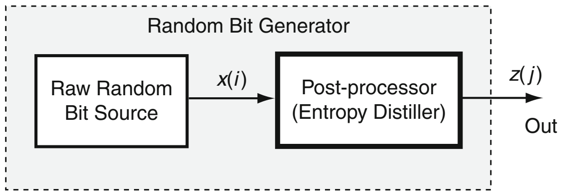
In the applications, the random source can be constructed of dedicated hardware devices; otherwise, the random source can use software procedures to extract random processes from the platform on which the generator is implemented.

Generators of the first type are commonly called hardware-based (HW); generators of the second type are software-based (SW).

SW-Based Generators

Generally, SW-based generators are implemented on computer systems and the values typically exploited as raw stream sources are obtained from:

- Event timings:
  - Mouse movements and clicks
  - Keystrokes
  - Disk and network accesses



Random Bit Generator. Fig. 1 Random bit generator

- Data depending on the history of the system and/or on a large amount of events:
  - System clock
  - I/O buffers
  - Load or network statistics

It is easy to understand that these types of sources are far from ideal. Entropy is mostly low and difficult to evaluate as well as the actual robustness with respect to observation and manipulation. SW-based generators should use more than one source, in order to be protected from the possibility that one or more sources could be compromised. Missing statistic evaluation, the post-processing should be planned assuming that the source entropy is very low; it should therefore execute a drastic compression and use a robust hash algorithm.

HW-Based Generators

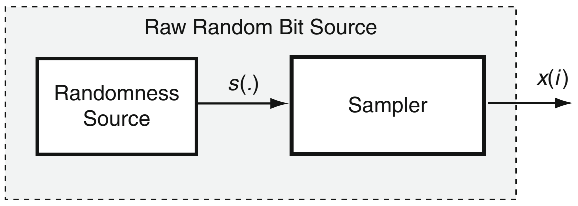
HW-based generators present the advantage of having a clearer model of the source and of its possible interaction with the outside. Generally, HW-based generators can be implemented using the common integrated technologies, and can be inserted in tamper resistant protections, in order to be protected from observation and manipulation. Normally, the nature of these sources (faster, higher in quality, and more protected) lowers the need of prostprocessing permitting to obtain a much higher throughput than the one obtainable from SW-based generators.

Typically the raw random stream source is a system that generates sequence$X$ by sampling and quantizing an analog nondeterministic value S (Fig. 2). Two possible quantization modes which represent two very common cases can be considered:

- A sign mode $x(i) = \text{sign}(s(i))$
- A mod 2 mode $x(i) = (s(i)) \bmod 2$

The quantization mod 2 (►Modular Arithmetic) presents the advantage of limiting or making negligible the effects of deterministic components of S. Let

$$S = a \cdot R + m + D$$



Random Bit Generator. Fig. 2 Random bit source

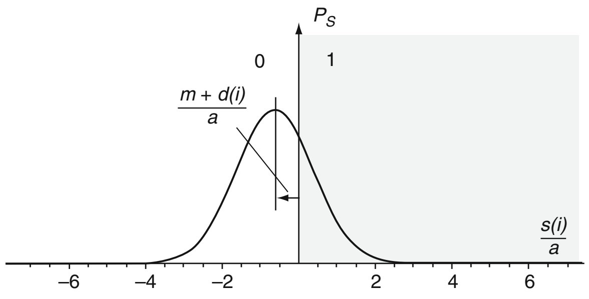
where R represents a normalized random process, a is an amplitude factor, m is an offset, and D represents a possible deterministic process. It can be observed that, no matter what is the amplitude of D and m, their effect becomes negligible when the amplitude of $a \cdot R$ is big enough.

As an example, Figs. 3 and 4 represent the case in which R has a normalized Gaussian distribution, $a = 0.5$, and $(m + d(i))/a = 0.6$. The x axis is divided into 0 and 1 zones to show how the R distribution is partitioned between 0 and 1 value samples. The mod 2 quantization mode results in 0 and 1 bands as shown in Fig. 4. Of course, in actual implementations, these bands can be asymmetric and this issue must be taken into account as possible cause of offset on the sequence $x(i)$.

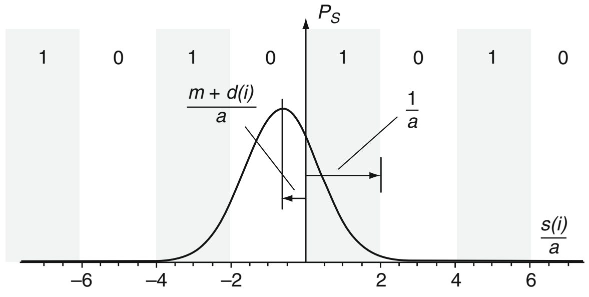
Randomness Sources in HW-Based Generators

Random sources commonly used by HW-based generators use phenomena such as:

- Electronic noises (thermal, shot, avalanche)
- Phase noises
- Flip-flop metastabilities



Random Bit Generator. Fig. 3 Normalized Gaussian distribution with $a = 0.5$



Random Bit Generator. Fig. 4 Normalized Gaussian distribution with $(m + d(i))/a = 0.6$

All these types of sources can be implemented by means of standard electronic devices.

Electronic Noise Sources. Among electronic noises, both the thermal noise in a resistor and the junction shot noise offer the advantage of being white noises with a Gaussian distribution whose intensity depends on physical values easy to keep under control. In fact, in the case of thermal noise, the power density of the noise depends merely on the resistor value and the absolute temperature, while in the case of the shot noise, the power density of the noise depends only on the current that flows through the junction. These phenomena make possible the realization of sources endowed of a simply analytic statistical model, instead of based on an empiric model depending on several technological and implementation factors. The deriving advantages are the following:

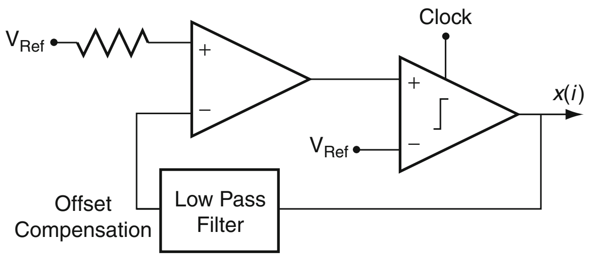
- The possibility to evaluate a priori source entropy and defects, and consequently the possibility to design a suitable post-processing
- Availability of a statistical model useful for verifying that the source works correctly

In Fig. 5, see the general scheme of a raw random bit source based on direct amplification technique. The noise's source consists of the resistor; the clocked comparator performs the sampling and the quantization of the amplified noise and the low pass feedback loop compensates the offsets due to the amplifier and the comparator.

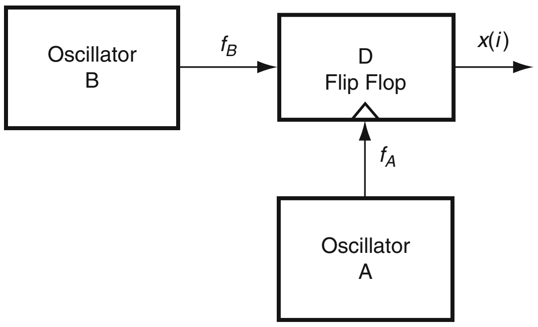
This kind of source can have a very high throughput. The only limitation is the bandwidth of the noise amplifier, given that as the sampling frequency increases, so does the correlation among samples. In fact, the sampling frequency can reach the amplifier high pass cutoff frequency since in any case the post-processing must be designed to be able to remove the stronger correlation that can result from an attack.

On the other hand, this design requires an accurate implementation. In fact the high-gain amplification needed for the thermal noise involves a critical offset compensation and a high sensitivity to internal or external interfering signals. Since the sign mode quantization is used, both an excess in the offset m or in an interfering signal D can block or force the source (see Fig. 3).

Phase Noise Sources. Generally, sources based on phase noise have a simple and robust implementation. In Fig. 6, a basic and typical scheme is depicted: a slow oscillator A samples a fast oscillator B. The D type flip-flop performs the mod 2 sampling of the phase difference between the two oscillators. Width and symmetry of the quantization bands depend respectively on the frequency and duty cycle of the oscillator B.



Random Bit Generator. Fig. 5 The direct amplification technique



Random Bit Generator. Fig. 6 A slow oscillator (A) sampling a fast oscillator (B)

In fact this scheme has an intrinsic periodic behavior due to the phase shifting that always occurs if f_B/f_A is not an integer. This effect is negligible if the phase noise is large with respect to half the period of the sampled oscillator (i.e., if in Fig. 4, noise distribution overflows the width of quantization bands).

More generally, since the mod 2 quantization is used, this device is robust with respect to any phase process that could be superimposed on the phase noise.

Basically the obtainable throughput depends on the f_B value and on the intrinsic noise of the oscillators. In fact, once f_B is maximized (i.e., quantization bands are made as narrow as possible), the sampling period $1/f_A$ must be long enough in order to accumulate a sufficient phase noise between two subsequent samples. Hence, noisier oscillators allow a higher sampling frequency.

A more efficient phase noise exploitation can be obtained by means of a phase control that allows sampling the output of oscillator B on its edges. In this way, even a little amount of phase noise is sufficient to get a random output. Basically this solution implies a phase control that plays the same role than the offset compensation in Fig. 5 scheme.

Generally, the statistical model of phase noises is not known a priori, being determined by several technological and implementation factors. Anyhow, in some solutions, an electronic noise source can be used to cause phase noises in an oscillator. In that way, it is possible to obtain a source whose phase noise characterization is known a priori since it is directly derived from an electronic noise.

Flip-Flop Metastability Sources. Actually, the random sources based on flip-flop metastability also exploit electronic and phase noises. Flip-flop metastability occurs when input signals are very close to the threshold and/or when data and clock signals switch very close in time to one another. In this condition, a small variation in levels or in phases results in a different output value.

The main implementation issue is the control of level or phase that is needed to give rise to metastability. Basically these are the same offset and phase control functions that are involved in the implementation of sources based on electronic and phase noises.

Post-Processing

In practice, every kind of raw random stream source can present defects as offset, auto-correlation or cross-correlation with other phenomena. Auto-correlation can occur because of the limitation in frequency bandwidth that is intrinsic in any physical randomness source. Other defects can be done to implementation issues as:

- Electrical or timing offsets (e.g., amplifier offset in Fig. 5 design, unbalanced duty cycle of the sampled oscillator in Fig. 6 design)
- Intrinsic design behaviors (e.g., in Fig. 5 design, the offset compensation suppresses long sequences of equal symbols; in Fig. 6 design, a lack of phase noise results in a periodic behavior)
- Interference with internal or external signals (e.g., power supply fluctuation, clock signals, switching signals coupled via the silicon substrate or via power supply, etc.)

However, the most important problems can be caused by manipulation attempts. For instance, an attacker trying to inject a signal D, wanting to force the exit $x(i)$.

The design of random sources, which are intrinsically very stable and robust with respect to these problems, results inevitably in a complex design and in drastic penalties on the performance. Moreover, it seems quite impossible, especially in consumer applications, to design a source that could resist a well-equipped attacker. A more effective approach seems to be the use of reasonably good random sources together with a suitable post-processing. According to the quality and robustness of the source, the post-processing can be based on simple scrambling and mixing techniques, or even on an actual hash algorithm.

Note the fact that an effective post-processing unavoidably hides the defects of the source, even when they are so massive to completely compromise the entropy of the generator. Practically, the more the entropy of the source tends to zero, the more the post-processor tends to act as if it were a pseudorandom generator.

This behavior makes it impossible to check the malfunction of the source through statistical tests applied to the output of the generator. On the contrary, most of the possible source anomalies can be revealed by means of statistical tests applied directly at the source, that is, before the behavior of the source is masked by post-processing and quantization. Obviously, the source must have a statistical characterization and a behavior that allow distinguishing normal defects and variances with respect to faults malfunction and attacks.

Depending on the kind of source, testing can even be very simple. As an example, in the device of Fig. 5, defects such as decreasing of the noise frequency bandwidth and/or of the amplification as well as increasing of the offset, all can be revealed by the decrease in the number of transition of $x(i)$.

A simple test performed on the amplitude of S (i.e., before sampling and quantization) can also reveal the attempt to force S by means of the superimposition of a known signal D. In fact this will inevitably result in an increase of the S amplitude. It must be noticed that the attacker can use a signal D such that the statistic of $x(i)$ is not changed. This means that, after quantization, since the amplitude information is removed, no statistical test can reveal this kind of attack.

Due to their simplicity, tests on the source can be executed even continually during generator operation. This allows to perform real-time check of malfunctions or even to dynamically tune the amount of post-processing compression depending on the estimated source quality.

Recommended Reading

1. Bagini V, Bucci M (1999) A design of reliable true random number generator for cryptographic applications. In: Koç ÇK, Paar C (eds) Proceedings of workshop cryptographic hardware and embedded systems, CHES '99, Worcester. Lecture Notes in Computer Science, vol 1717. Springer, Berlin, pp 204–218
2. Bendat JS (1958) Principles and applications of random noise theory. Wiley, New York

3. Davis D, Ihaka R, Philip F (1994) Cryptographic randomness from air turbulence in disk drives. In: Desmedt Y (ed) Advances in cryptology – CRYPTO'94. Lecture Notes in Computer Science, vol 839. Springer, Heidelberg, pp 114–120
4. Dichtl M, Janssen N (2000) A high quality physical random number generator. In: Proceedings of Sophia Antipolis forum microelectronics (SAME 2000), Sophia-Antipolis, pp 48–53
5. FIPS 140-1 (1994) Security requirements for cryptographic modules. National Institute of Standards and Technology, GPO, Washington, DC
6. Gude M (1985) Concepts for a high performance random number generator based on physical random phenomena. Frequenz 39(7–8):187–190
7. Holman WT, Connelly JA, Downlatabadi AB (1997) An integrated analog/digital random noise source. IEEE Trans Circuits Syst I 44(6):521–528
8. Jun B, Kocher P (1999) The Intel random number generator. Cryptography Research Inc., white paper prepared for Inter Corp. http://www.cryptography.com/resources/whitepapers/Inte-lRNG.pdf
9. Knuth DE (1981) The art of computer programming, 2nd edn. Addison-Wesley, Reading
10. Maddocks RS, Matthews S, Walker EW, Vincent CH (1972) A compact and accurate generator for truly random binary digits. J Phys E5(8):542–544
11. Memezes AJ, Oorschot PC, Vanstone SA (2001) Handbook of applied cryptology. CRC, Boca Raton
12. Murry HF (1970) A general approach for generating natural random variables. IEEE Trans Comput C-19:1210–1213
13. Papoulis A (1965) Probability, random variables and stochastic processes. McGraw-Hill, New York
14. Petrie CA (1997) An integrated random bit generator for applications in cryptography. PhD thesis, Georgia Institute of Technology, Atlanta
15. Petrie CS, Connelly JA (1996) Modeling and simulation of oscillator-based random number generators. In: Proceedings of IEEE international symposium on circuits and systems, ISCAS'96, vol 4. IEEE, Atlanta, pp 324–327
16. Petrie CS, Connelly JA (2000) A noise-based IC random number generator for applications in cryptography. IEEE Trans Circuits Syst I 47(5):615–621
17. Schneier B (1996) Applied cryptography, 2nd ed. Wiley, New York
18. Trichina E, Bucci M, De Seta D, Luzzi R (2001) Supplementary cryptographic hardware for smart cards. IEEE Micro 21(6): 26–35
19. Vincent CH (1970) The generation of truly random binary numbers. J Phys E3(8):594–598
20. Vincent CH (1971) Precautions for the accuracy in the generation of truly random binary numbers. J Phys E4(11): 825–828

Random Key Predistribution

►Probabilistic Key Sharing

Random Number Testing

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

Related Concepts

►Entropy; ►Random Number Generation

Definition

Random values are critical to the ability of cryptography to resist crypto analysis and attacks. Testing the randomness of the cryptographic key string is a valuable and critical activity to determining the quality of the implementation.

Background

Random numbers generators implementations are generally in one of two forms, a random number generator, RNG or a pseudorandom number generator, PRNG. A random number generator typically uses hardware to simulate a random event and establish a random value. A pseudorandom number generator is a cryptographic algorithm implementation that by design outputs a string of bits that has random characteristics, although the output has very good randomness characteristics, there is a deterministic process to get from one random value to the next.

Because of the critical nature of Random Numbers, it is import for cryptographic implementations to test both the operation of the RNG or PRNG at start-up and during use to ensure that it is functioning and also perform a separate battery of tests when the design is implemented to confirm the design functions correctly. The design tests are much more comprehensive while the operational tests are oriented to detecting health of the implementation.

Theory/Application

Bits are in one of two states, zero or one, similar to the two sides of a coin, heads or tails. For a random bit or number generator each bit is independently chosen to be a one or zero, and has no relation to the either to previous or the following bit value, similar to a coin toss. This independence is a critical characteristic in providing cryptographic strength. Any patterns or predictability in the randomness of the bit states weakens the cryptographic strength as predictability can be analyzed and exploited.

A typical operational test is to verify that the random number does not repeat. This technique stores the last value or series of values output from the RNG/PRNG and compares it with the current one. If the value is repeated, it is discarded and a new value is created, if this same number is repeated multiple times, a fatal error is recognized as the random number generator has ceased to function.

Testing the design of a new random number generator is typically a series of tests, each of which evaluates specific characteristics of the bit/number strings. A quality random number has many individual features making one test to measure the quality of the implementation infeasible. There are a great many tests to evaluate individual parameters but typically there are a few (~20) which are typically evaluated by testing, some of those parameters and the associated tests are described in the paragraphs below.

Another concept that is important to consider is the number of bits that are available for testing. The evaluation of RNG/PRNGs depends on obtaining a statistically significant sample of bits, which may typically be large quantity and may take significant time to generate the quantity of data required. All of the testing done on the RNG/PRNGs is statistical in nature and as such the larger the sample the more relevant the results. It should be noted that as this are statistical tests and it is not possible to assign defined pass and fail criteria, and in some cases a result that is not typical, may not indicate a clear failure. Four of the easiest and first tests to be run are included below.

The Monobit or Frequency Test for Entire Sequence

The purpose of the frequency test is to determine the proportion of zeroes and ones in the sequence. The purpose is to determine if the quantity of ones in a sequence are approximately the same as the number of zeros. This is typically the first test to be performed because if this test does not pass the other tests are not expected to pass either.

The Runs Test for the Entire Sequence

The purpose is to determine the total number of runs (uninterrupted sequence of identical bits) in the sequence. This test assesses whether the change between zeros and ones is too fast or too slow.

Often cryptography is performed on specific block lengths of data; therefore, it becomes relevant to evaluate the characteristics of the data string when observed in blocks similar to what will be used by the algorithm.

Test for the Longest-Run-of-Ones in a Block

The purpose is to determine the total number of runs (uninterrupted sequence of identical bits) in a block. This test assesses whether the change between zeros and ones is too fast or too slow.

Frequency Test Within a Block

The purpose of the test is the proportion of ones within the block. The purpose t is to assess whether the frequency of ones, as would be approximately half of the block size.

Open Problems

Two primary challenges confront the tester.

- The testing is complex and the tools are difficult to use.
- The need for large sample size of random bits can often be time consuming and complex to collect.

Experimental Results

The tests are well established and vetted therefore, results are

Recommended Reading

1. NIST Special Publication SP800-22B A statistical test suite for random and pseudorandom number generators for cryptographic applications

Random Oracle Model

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

Related Concepts

►Cryptographic Hash Function; ►Pseudorandom Function; ►Public Key Encryption; ►Random Oracle Model and Undeniable Signatures; ►Randomness

Definition

Many cryptographic algorithms and protocols use ►pseudo-random functions as computationally efficient approximations of random values. It is theoretically clear that any instantiation of a pseudo-random function can NOT return a truly random value, but rather one that is more or less unpredictable. When we try to formally prove the security of a given algorithm or protocol, we are faced with the question if any particular pseudo-random function might unluckily interfere with the rest of the algorithm or protocol thereby incorporating a security weakness. The random oracle model is a way to modularize such formal security proofs by assuming that all pseudo-random used by an algorithm or protocol were replaced by random oracles, which return truly random values upon invocation.

The resulting security proof is then to assure that the algorithm or protocol is at least secure "modulo" the way it makes its random choices.

Theory

The random oracle model was introduced by Bellare and Rogaway [2]. The idea is a simple one: namely provide all parties of a ►protocol – good and bad alike – with access to a (public) function *h* and then prove the protocol to be correct assuming that *h* maps each input to a truly random output, i.e., it behaves like a truly random oracle. Later, in practice, one sets *h* to some specific function derived in some way from a standard cryptographic ►hash function like ►SHA-1 [5], ►MD5 [6], ►RIPEMD-160 [4], or others. It is clear though that any specific function will not be random because it is deterministic, i.e., it returns the same value when given the same input. (Also see Bellare's overview of the random oracle model in [1].) The random oracle model buys efficiency and, as Rogaway claims, security guarantees, which, although not at the same level as those provided by the standard "provable security approach," are arguably superior to those provided by a totally ad hoc protocol design.

The overly skeptical might say that a security proof in the random oracle model gains nothing because the function *h* that one actually uses in the final protocol is not random. Here is another way to look at it. In practice, attacks on schemes involving a hash function *h* derived from SHA-1 and ►number theory will often themselves treat *h* as random. Bellare and Rogaway call such attacks *generic*. In other words, cryptanalysis of these "mixed" protocols is usually done by assuming *h* to be random. But then proofs in the random oracle model apply and indeed show that such generic attacks will fail unless the underlying number-theoretic problems are easy to solve. In other words, the analysis at least provably excludes a certain common class of attacks, namely generic ones.

It is important to choose carefully the instantiating function *h*. The intuition stated by Bellare and Rogaway in [2] is that the resulting protocol is secure as long as the protocol and the hash function are sufficiently "independent," which means the protocol does not itself refer to the hash function in some way. This is a fuzzy guideline that needs more work in the future.

An important step in better understanding the random oracle model was taken by Canetti et al. [3]. They show there exists protocols secure in the random oracle model, but insecure under any instantiations in which we substitute a function from a small family of efficiently computable functions. Their examples, however, are somewhat contrived, and this kind of situation does not arise with any of the "real" cryptographic mechanisms in the literature.

Open Problems and Future Directions

In comparison with a totally ad hoc design, a proof in the random oracle model has the benefit of judging the protocol under a strong, formal notion of security, even if this assumes some underlying primitive to be very strong. This is better than not formally modeling the security of the protocol at all. This explains why the random oracle model is viewed as a "bridge" between theory and practice [2].

Recommended Reading

1. Bellare M (1999) Practice-oriented provable security. In: Lectures on data security. Lecture notes in computer science, vol 1561. Springer, Berlin, pp 1–15
2. Bellare M, Rogaway P (1993) Random oracles are practical: a paradigm for designing efficient protocols. In: First ACM Conference on Computer and Communications Security, Proceedings, Fairfax, Nov 1993. ACM Press, New York, pp 62–73
3. Canetti R, Goldreich O, Halevi S (1998) The random oracle methodology, revisited. In: 30th symposium on theory of computing (STOC) 1998. ACM Press, New York, pp 209–218
4. Dobbertin H, Bosselaers A, Preneel B (1996) RIPEMD-160: a strengthened version of RIPEMD. In: Gollman D (ed) Fast software encryption, third international workshop, Cambridge, UK, 21–23 Feb 1996, Proceedings. Lecture notes in computer science, vol 1039. Springer, Berlin, pp 71–82
5. National Institute of Standards and Technology (NIST) (1995) Secure hash standard. Federal Information Processing Standards Publication (FIPS PUB 180–1)
6. Rivest R (1992) The MD5 message-digest algorithm; RFC1321. http://www.faqs.org/rfcs/rfc1321.html

Rank Codes

ERNST M. GABIDULIN
Department of Radio Engineering, Moscow Institute of Physics and Technology (State University), Dolgoprudny, Moscow region, Russia

Synonyms

Rank-metric codes

Related Concepts

►Algebraic Coding Theory; ►Code-Based Cryptography; ►McElice Public Key Cryptosystem

Definition

There exist two representations of Rank codes: *matrix* representation and *vector* representation.

In *matrix* representation, rank codes are defined as subsets of a normed space $\left\{\mathbb{F}_q^{N\times n},\ \mathrm{Rk}\right\}$ of $N \times n$ matrices over a finite (base) field F_q, where the norm of a matrix $M \in \mathbb{F}_q^{N\times n}$ is defined to be the algebraic rank $\mathrm{Rk}(M)$ of this matrix over $\mathbb{F}_q$. The *rank distance* between two matrices M_1 and M_2 is the rank of their difference $\mathrm{Rk}(M_1 - M_2)$. The *rank distance* of a matrix rank code $\mathcal{M} \subset \mathbb{F}_q^{N\times n}$ is defined as the minimal pairwise distance: $d(\mathcal{M}) = d = \min(\mathrm{Rk}(M_i - M_j) : M_i, M_j \in \mathcal{M},\ i \neq j)$.

In *vector* representation, rank codes are defined as subsets of a normed n-dimensional space $\left\{\mathbb{F}_{q^N}^n,\ \mathrm{Rk}\right\}$ of n-vectors over an extension field $\mathbb{F}_{q^N}$, where the norm of a vector $\mathbf{v} \in \mathbb{F}_{q^N}^n$ is defined to be the *column* rank $\mathrm{Rk}(\mathbf{v} \mid \mathbb{F}_q)$ of this vector over $\mathbb{F}_q$, that is, the maximal number of coordinates of $\mathbf{v}$ which are linearly independent over the base field $\mathbb{F}_q$. The *rank distance* between two vectors $\mathbf{v}_1$, $\mathbf{v}_2$ is the column rank of their difference $\mathrm{Rk}(\mathbf{v}_1 - \mathbf{v}_2 \mid \mathbb{F}_q)$. The *rank distance* of a vector rank code $\mathcal{V} \subset \mathbb{F}_{q^N}^n$ is defined as the minimal pairwise distance: $d(\mathcal{V}) = d = \min(\mathrm{Rk}(\mathbf{v}_i - \mathbf{v}_j) : \mathbf{v}_i, \mathbf{v}_j \in \mathcal{V},\ i \neq j)$.

Background

Algebraic coding theory may be considered as the theory of subsets of a certain *normed* finite-dimensional space Γ over the finite field equipped with a norm function $\mathfrak{N}$. The most known norm in coding theory is the *Hamming weight* of a vector. It turns out that the rank function $\mathrm{Rk}(A)$ of matrices A over fields can be considered as the norm function. In particular, the well-known inequalities for sums of matrices $|\mathrm{Rk}(A) - \mathrm{Rk}(B)| \leq \mathrm{Rk}(A+B) \leq \mathrm{Rk}(A) + \mathrm{Rk}(B)$ define implicitly the rank distance relations on the space of all matrices of identical size. Explicitly, the concept of the rank metric was introduced by Loo-Keng Hua [1] as "arithmetic distance." Philippe Delsarte [2] defined the rank distance (or, q-distance) on the set of bilinear forms (equivalently, on the set of rectangular matrices)and proposed the construction of optimal codes in *bilinear form* representation. Ernst M. Gabidulin [3] introduced the rank distance for vector spaces over extension fields and found connections between rank codes in the *vector* representation and in the *matrix* representation. Optimal codes in *vector* representation were described. Fast coding and decoding algorithms were proposed for optimal codes.

Theory

The normed spaces $\left\{\mathbb{F}_q^{N\times n},\ \mathrm{Rk}\right\}$ and $\left\{\mathbb{F}_{q^N}^n,\ \mathrm{Rk}\right\}$ are isomorphic isometrically. Let a basis $\Omega = \{\omega_1, \omega_2, \ldots, \omega_N\}$ of $\mathbb{F}_{q^N}$ over $\mathbb{F}_q$ be chosen. Then each vector $\mathbf{v} = \begin{bmatrix} v_1 & v_2 & \ldots & v_n \end{bmatrix} \in \mathbb{F}_{q^N}^n$ can be mapped into the $N \times n$ matrix $M \in \mathbb{F}_q^{N\times n}$ by replacing each coordinate v_j with the N-column consisting of coefficients in representing v_j by the basis Ω. This mapping is bijective and isometric.

Given a rank code $\mathcal{M}$ in matrix representation, one can construct a rank code $\mathcal{V}$ in vector representation with the same size, code distance, and pairwise distances, and vice versa.

The size $|\mathcal{M}| = |\mathcal{V}|$ of related codes with code distance d satisfy the Singleton bound $|\mathcal{M}| = |\mathcal{V}| \leq \min(q^{N(n-d+1)}, q^{n(N-d+1)})$. Codes reaching this bound are called maximum rank distance codes, or, MRD codes.

A rank code $\mathcal{M}$ in matrix representation is called $\mathbb{F}_q$*-linear* if $\mathcal{M}$ is a subspace of $\mathbb{F}_q^{N\times n}$.

A rank code $\mathcal{V}$ in vector representation is called $\mathbb{F}_{q^N}$*-linear* if $\mathcal{V}$ is a subspace of $\mathbb{F}_{q^N}^n$.

Mapping a $\mathbb{F}_{q^N}$*-linear* code $\mathcal{V}$ in vector representation into related code $\mathcal{M}$ in matrix representation results in a $\mathbb{F}_q$*-linear* code.

Mapping a $\mathbb{F}_q$*-linear* code $\mathcal{M}$ in matrix representation into related code $\mathcal{V}$ in vector representation results in *not necessary* a $\mathbb{F}_{q^N}$*-linear* code.

Constructions of $\mathbb{F}_q$*-linear* rank codes in the matrix representation and $\mathbb{F}_{q^N}$*-linear* rank codes in the vector representation will be considered.

Delsarte's Optimal Rank Codes in Matrix Representation

Delsarte's construction of rank codes in bilinear form representation is presented here in matrix representation.

Assume that $n \leq N$. Let $\mathrm{Tr}(x) = \sum_{l=0}^{N-1} x^{q^l}$, $x \in \mathbb{F}_{q^N}$, be the trace function from $\mathbb{F}_{q^N}$ into $\mathbb{F}_q$. Let d be an integer in $\{1, 2, \ldots, n\}$. Let $\mathbf{u} = \begin{bmatrix} u_0 & u_1 & \ldots & u_{n-d} \end{bmatrix} \in \mathbb{F}_{q^N}^{n-d+1}$. Let $\mu_1, \mu_2, \ldots, \mu_n$ be linearly independent elements of $\mathbb{F}_{q^N}$. Let $\Omega = \{\omega_1, \omega_2, \ldots, \omega_N\}$ be a basis for $\mathbb{F}_{q^N}$.

Define a code in matrix representation as the set of $N \times n$ matrices $\mathcal{M} = \left\{M(\mathbf{u}) = [M_{ij}(\mathbf{u})] : \mathbf{u} \in \mathbb{F}_{q^N}^{n-d+1}\right\}$, where

$$M_{ij}(\mathbf{u}) = \mathrm{Tr}\left(\sum_{s=0}^{n-d} u_s \omega_i \mu_j^{q^s}\right).$$

Then $\mathcal{M}$ is a rank code with code distance d reaching the Singleton bound $|\mathcal{M}| = q^{N(n-d+1)}$.

Let $A_i(n,d)$, $i = 0, 1, \ldots, n$, be the number of code matrices with rank i. The weight distribution is as follows:

$$A_i(n,d) = \begin{cases} 1, & \text{if } \quad i = 0; \\ 0, & \text{if } \quad i = 1, \ldots, d-1; \\ \begin{bmatrix} n \\ i \end{bmatrix} \sum_{s=0}^{i-d} (-1)^s \begin{bmatrix} i \\ s \end{bmatrix} q^{\frac{s(s-1)}{2}} (q^{N(d-i+1-s)} - 1), & \text{if } \quad i = d, \ldots, n, \end{cases}$$

where $\begin{bmatrix} n \\ i \end{bmatrix} = \prod_{j=0}^{i-1} \frac{q^n - q^j}{q^i - q^j}$ is the *Gaussian binomial coefficient*.

Optimal Rank Codes in Vector Representation

A $\mathbb{F}_{q^N}$-linear vector code $\mathcal{V}$ is a subspace of the normed space $\left\{\mathbb{F}_{q^N}^n, \text{Rk}\right\}$. Denote by (n,k,d) a code $\mathcal{V}$ of dimension $k \le n$ and rank distance d. Such a code can be described in terms of a full rank *generator* matrix G_k over the extension field F_{q^N} of size $k \times n$. Code vectors $\{\mathbf{v}\}$ are all linear combinations of this matrix. Thus, the size of a code is equal to $|\mathcal{V}| = q^{Nk}$.

Equivalently, a rank code $\mathcal{V}$ can be described in terms of a full rank *parity-check* matrix H_{n-k} over $\mathbb{F}_{q^N}$ of size $(n-k) \times n$. It satisfies the condition $G_k H_{n-k}^\top = O$, where O is the all zero $k \times (n-k)$ matrix. Code vectors $\{\mathbf{v}\}$ are all solutions of the linear system of equation $\mathbf{v}H_{n-k}^\top = \mathbf{0}$.

For optimal (MRD) codes, it must be $k = n - d + 1$, or, $n - k = d - 1$.

General constructions of MRD codes in terms of parity-check matrices can be described as follows. Let $h_1, h_2, \ldots, h_n$ be a set of elements from the extension field $\mathbb{F}_{q^n}$ *linearly independent* over the base field $\mathbb{F}$. Let s be a positive integer such that $\gcd(s,N) = 1$. Then, a parity matrix of the form

$$H_{d-1} = \begin{bmatrix} h_1 & h_2 & \ldots & h_n \\ h_1^{q^s} & h_2^{q^s} & \ldots & h_n^{q^s} \\ h_1^{q^{2s}} & h_2^{q^{2s}} & \ldots & h_n^{q^{2s}} \\ \ldots & \ldots & \ldots & \ldots \\ h_1^{q^{(d-2)s}} & h_2^{q^{(d-2)s}} & \ldots & h_n^{q^{(d-2)s}} \end{bmatrix}.$$

defines an MRD (n,k,d) code with code length $n \le N$, dimension $k = n - d + 1$ and rank distance $d = n - k + 1$.

Equivalently, general constructions of MRD codes can be described in terms of generator matrices. Let $g_1, g_2, \ldots, g_n$ be a set of elements from the extension field $\mathbb{F}_{q^n}$ *linearly independent* over the base field $\mathbb{F}$. Then, a generator matrix of the form

$$G_k = \begin{bmatrix} g_1 & g_2 & \ldots & g_n \\ g_1^{q^s} & g_2^{q^s} & \ldots & g_n^{q^s} \\ g_1^{q^{2s}} & g_2^{q^{2s}} & \ldots & g_n^{q^{2s}} \\ \ldots & \ldots & \ldots & \ldots \\ g_1^{q^{(k-1)s}} & g_2^{q^{(k-1)s}} & \ldots & g_n^{q^{(k-1)s}} \end{bmatrix}.$$

defines an MRD (n,k,d) code with code length $n \le N$, dimension $k = n - d + 1$ and rank distance $d = n - k + 1$.

The weight distribution of vector MRD codes coincides for a given d with the weight distribution of Delsarte's codes above.

The case $s = 1$ is used mostly.

No other constructions of MRD codes are known (2009).

Correcting Rank Errors and Rank Erasures

Let a MRD $(n,k,d = n-k+1)$ code $\mathcal{V}$ be given. Let a transmitted signal be $\mathbf{v}$ and received signal be $\mathbf{y} = \mathbf{v} + \mathbf{e}_{\text{total}}$, where $\mathbf{e}_{\text{total}}$ is an error. The code $\mathcal{V}$ can correct *in general* vector errors of the form

$$\begin{aligned} \mathbf{e}_{\text{total}} &= \mathbf{e} + \mathbf{e}_{\text{row}} + \mathbf{e}_{\text{col}} \\ &= e_1\mathbf{u}_1 + e_2\mathbf{u}_2 + \cdots + e_t\mathbf{u}_t + \\ &\quad + a_1\mathbf{r}_1 + a_2\mathbf{r}_2 + \cdots + a_v\mathbf{r}_v + \\ &\quad + w_1\mathbf{c}_1 + w_2\mathbf{c}_2 + \cdots + w_l\mathbf{c}_l \end{aligned}$$

provided that $2t + v + l \le d - 1$.

The part $\mathbf{e} = e_1\mathbf{u}_1 + e_2\mathbf{u}_2 + \cdots + e_t\mathbf{u}_t$ is called a *random rank error* of rank t under assumption that elements $e_i \in \mathbb{F}_{q^N}$ are linearly independent over the base field $\mathbb{F}_q$ and *unknown* to the decoder; n-vectors $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_t$ have coordinates in the *base* field $\mathbb{F}_q$, are linearly independent over the base field $\mathbb{F}_q$ and also *unknown* to the decoder. The rank t is *unknown* to the decoder.

The part $\mathbf{e}_{\text{row}} = a_1\mathbf{r}_1 + a_2\mathbf{r}_2 + \cdots + a_v\mathbf{r}_v$ is called a *vector rank* ***row*** *erasure with side information* under assumption that elements $a_i \in \mathbb{F}_{q^N}$ are linearly independent over the base field $\mathbb{F}$ and *known* to the decoder; n-vectors $\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_v$ have coordinates in the *base* field $\mathbb{F}_q$, are linearly independent over the base field $\mathbb{F}_q$ and *unknown* to the decoder.

The part $\mathbf{e}_{\text{col}} = w_1\mathbf{c}_1 + w_2\mathbf{c}_2 + \cdots + w_l\mathbf{c}_l$ is called a *vector rank* ***column*** *erasure with side information* under assumption that elements $w_i \in \mathbb{F}_{q^N}$ are linearly independent over the base field $\mathbb{F}_q$ and are *unknown* to the decoder; n-vectors $\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_l$ have coordinates in the base field $\mathbb{F}_q$, are linearly independent over the base field $\mathbb{F}_q$ and *known* to the decoder.

First fast-correcting random rank errors only was proposed in [3]. The algorithm is based on the extended Euclidean division algorithm for linearized polynomials. There exist several further modifications.

Algorithms for correcting random rank errors and rank erasures simultaneously are proposed in [4–6].

Open Problems

For the Hamming metric and generalized Reed–Solomon codes, there exists the *Sudan–Guruswami algorithm* correcting random errors beyond the $(d-1)/2$ bound.

An open problem: does similar algorithm exist for the rank metric and MRD codes?

Recommended Reading

1. Hua L-K (1951) A theorem on matrices over a field and its applications. Chin Math Soc 1(2):109–163
2. Delsarte P (1978) Bilinear forms over a finite field, with applications to coding theory. J Comb Theory A 25:226–241

3. Gabidulin EM (1985) Theory of codes with maximum rank distance. Probl Inf Transm 21(1):1–12
4. Gabidulin EM, Paramonov AV, Tretjakov OV (1992) Rank errors and rank erasures correction. In: Proceedings of the 4th international colloquium on coding theory, 30 September–7 October 1991, Dilijan, Armenia, pp 11–19, Yerevan, 1992
5. Gabidulin EM, Pilipchuk NI (2008) Error and erasure correcting algorithms for rank codes. Designs Codes Cryptogr 49:105–122. DOI 10.1007/s10623-008-9185-7
6. Silva D, Kschischang FR, Koetter R (2008) A rank-metric approach to error control in random network coding. IEEE Trans Inf Theory 54(9):3951–3967

Rank-Matric Codes

►Rank Codes

RBAC

►Roles in SQL
►Role-Based Access Control

RC4

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

Related Concepts

►Sequences; ►Stream Cipher; ►Synchronous Stream Cipher

Definition

RC4 is a symmetric ►synchronous stream cipher. It is initialized by a variable length key, between 40 and 128 bits. The key is used to compute the initial value of the internal state of a pseudorandom generator, which produces a keystream. Encryption (respectively decryption) then consists in XOR-ing this keystream to the plaintext (respectively the ciphertext).

The internal state of the pseudorandom generator consists of a permutation *S* of all the 256 bytes, and two indices *i* and *j*. The secret key determines which permutation should be used as the initial state, before starting encryption/decryption. This step is called the key-scheduling. Then, the pseudorandom generator outputs as many bytes as needed to process encryption/decryption. The whole process is described in Fig. 1.

```
Setup S according to the secret key
i := 0
j := 0
while Needed
    i := (i + 1) mod 256
    j := (j + S[i]) mod 256
    swap the values of S[i] and S[j]
    U := S[(S[i] + S[j]) mod 256]
    output U
end
```

RC4. Fig. 1 The pseudorandom number generator used in RC4

Background

RC2 and RC4 are ciphers developed by R. Rivest for RSA Data Security, Inc. They are proprietary, and their details have not been officially published. RC2 is a variable-key-size block cipher, and RC4 a variable-key-size stream cipher (the key size may vary between 1 up to 2, 048 bits). RC4 is used a lot today; you can find it in Secure Socket Layer (SSL), in Wi-Fi security protocols, and so on. Some reverse engineering has been done, and an algorithm was accessible on the Internet in 1994, that gave the same output as RC4.

Theory

Several studies and attacks have been published since 1997. A good starting point on the subject is [1]. But several security flaws have been pointed out more recently.

A first point is that RC4 does not include any IV vector (refer the entry on ►synchronous stream ciphers for more details). This is a quite important drawback, that can be countered by some tricks, as combining RC4 use with some hash function, but remains a security weakness.

A weakness concerning the correlation between the key and the keystream has been pointed out in 2001 [2]. This weakness can be really dramatic when the key is not random, as in the WEP wi-fi authentication [3]. A more powerful attack exploiting such a correlation has been proposed in 2005 [4], which led to real attacks on WEP keys [5].

More recent papers pointed out different biases that can be used to reconstruct the secret key [6–11]. Some others also discuss how to distinguish RC4's output from a real random sequence [12–15], the last one showing how to relate the output to index *j*.

Moreover, some combinatorial weaknesses have been proved [16], and key collisions have been pointed out [17–19].

For all these reasons, RC4 cannot be considered as secure anymore.

Applications

RC4 is used a lot today; you can find it in Secure Socket Layer (SSL), in Wi-Fi security protocols (WEP, WPA), etc.

Recommended Reading

1. Mantin I (2001) Analysis of the stream cipher RC4. Master's thesis, Weizmann Institute of Science, Rehovot
2. Fluhrer S, Mantin I, Shamir A (2001) Weaknesses in the key scheduling algorithm of RC4. In: SAC'01, Toronto, August 2001. Lecture notes in computer science, vol 2259. Springer, Berlin, pp 1–24
3. Stubblefield A, Ioannidis J, Rubin AD (2001) Using the Fluhrer, Mantin, and Shamir attack to break WEP. AT&T Labs technical report TD-4ZCPZZ
4. Klein A (2008) Attacks on the RC4 stream cipher. Des Codes Cryptogr 48(3):269–286
5. Tews E, Weinmann R-P, Pyshkin A, Breaking 104-bit WEP in under a minute. IACR ePrint 2007/120. http://www.cdc.informatik.tu-darmstadt.de/aircrack-ptw/
6. Paul G, Rathi S, Maitra S (2008) On non-negligible bias of the first output byte of RC4 towards the first three bytes of the secret key. Des Codes Cryptogr 49(1–3):123–134
7. Paul G, Maitra S (2007) Permutation after RC4 key scheduling reveals the secret key. In: Selected areas in cryptography – SAC 2007, Ottawa, August 2007. Lecture notes in computer science, vol 4876. Springer, Berlin, pp 360–377
8. Biham E, Carmeli Y (2008) Efficient reconstruction of RC4 keys from internal states. In: Fast software encryption – FSE 2008, Lausanne, February 2008. Lecture notes in computer science, vol 5086. Springer, Berlin, pp 270–288
9. Akgun M, Kavak P, Demirci H (2008) New results on the key scheduling algorithm of RC4. In: INDOCRYPT 2008, Kharagpur, December 2008. Lecture notes in computer science, vol 5365. Springer, Berlin, pp 40–52
10. Basu R, Maitra S, Paul G, Talukdar T (2009) On some sequences of the secret pseudo-random index j in RC4 key scheduling. In: International symposium on applied algebra, algebraic algorithms and error correcting codes – AAECC 2009, Tarragona, June 2009. Lecture notes in computer science, vol 5527. Springer, Berlin, pp 137–148
11. Maitra S, Paul G (2008) New form of permutation bias and secret key leakage in keystream bytes of RC4. In: Fast software encryption – FSE 2008, Lausanne, February 2008. Lecture notes in computer science, vol 5086. Springer, Berlin, pp 253–269
12. Mantin I, Shamir A (2001) A practical attack on broadcast RC4. In: FSE'01, Yokohama, April 2001. Lecture notes in computer science, vol 2355. Springer, Berlin, pp 152–164
13. Paul S, Preneel B (2003) Analysis of non-fortuitous predictive states of the RC4 keystream generator. In: INDOCRYPT 2003, New Delhi, December 2003. Lecture notes in computer science, vol 2904. Springer, Berlin, pp 52–67
14. Fluhrer SR, McGrew DA (2000) Statistical analysis of the alleged RC4 key stream generator. In: FSE'00, New York, April 2000. Lecture notes in computer science vol 1978. Springer, Berlin, pp 19–30
15. Basu R, Ganguly S, Maitra S, Paul G (2008) A complete characterization of the evolution of RC4 pseudo random generation algorithm. J Math Cryptol 2(3):257–289
16. Paul S Preneel B (2004) A new weakness in the RC4 keystream generator and an approach to improve the security of the cipher. In: Fast software encryption – FSE 2004, Delhi, February 2004. Lecture notes in computer science, vol 3017. Springer, Berlin, pp 245–259
17. Matsui M (2009) Key collisions of the RC4 stream cipher. In: Fast software encryption – FSE 2009, Leuven, February 2009. Lecture notes in computer science, vol 5665. Springer, Berlin, p 41
18. Chen J, Kiyaji A (2010) Generalized RC4 key collisions and hash collisions. In: International conference on security and cryptography for networks – SCN'10, Amalfi, September 2010. Lecture notes in computer science, vol 6280. Springer, Berlin, pp 73–87
19. Chen J (2010) A new class of RC4 colliding key pairs with greater hamming distance. In: IPSEC 2010, Seoul, May 2010. Lecture notes in computer science, vol 6047. Springer, Berlin, pp 30–44

RC5

Helena Handschuh
Computer Security and Industrial Cryptography Research Group, Katholieke Universiteit Leuven, Leuven - Heverlee, Belgium

Synonyms

Rivest cipher 5; Ron's code 5

Related Concepts

►Block Ciphers; ►Symmetric Cryptography

Definition

RC5 is an iterative secret-key block cipher published by Rivest [6] in 1995. It has variable parameters such as the key size, the block size, and the number of rounds. A particular (parameterized) RC5 encryption algorithm is designated as RC5-$w/r/b$, where w is the word size (one block is made of two words), r is the number of rounds ($r = 2h$), and b is the number of bytes for the secret key. RC5 served as a model for the Advanced Encryption Standard (AES) candidate RC6.

Background

Block ciphers in 1995 followed one of two generic models: Either they were based on a Feistel structure, or on a substitution-permutation network. RC5 is no exception and is a Feistel-like structure. It uses very few, very simple, and very fast operations in its round function; it is one of the first block ciphers using variable parameters and data-dependent rotations. Another novelty of RC5 is that it incorporates a complex one-way key scheduling procedure.

Theory

The "nominal" choice for the algorithm, RC5-32/12/16, has a 64-bit block size, 12 rounds, and a 128-bit key. The secret key is first expanded into a table of $2h + 2$ secret words S_i of w bits according to the key schedule. Let (L_0, R_0) denote the left and right halves of the plaintext. Note that a w-bit word is equivalently viewed as an integer modulo 2^w (►Modular Arithmetic). Then the encryption algorithm is given by:

```
$L_1 \leftarrow L_0 + S_0 \bmod 2^w$
$R_1 \leftarrow R_0 + S_1 \bmod 2^w$
for $i = 1$ to $2h$ do
    $L_{i+1} \leftarrow R_i$
    $R_{i+1} \leftarrow ((L_i \oplus R_i) \lll R_i) + S_{i+1} \bmod 2^w$
```

where "$\oplus$" represents bit-wise exclusive-or, and "$X << Y$" is the (data-dependent) rotation of X to the left by the $\log_2 w$ least significant bits of Y. The ciphertext is (L_{2h+1}, R_{2h+1}), and each half-round i involves exactly one subkey S_i.

Applications

Kaliski and Yin [4] studied both differential and linear attacks on nominal RC5. Knudsen and Meier [5] further improved over their attacks and also showed that RC5 has weak keys. The best differential (chosen plaintext) and linear (known plaintext) attacks on RC5 are respectively the one by Biryukov and Kushilevitz [1] which breaks 12 rounds of RC5-32/12/16 with 2^{44} chosen plaintexts using partial differentials, and the one by Borst et al. [2] which breaks RC5 up to 10 rounds using multiple linear approximations. Both results suggest that RC5 should be used with at least 16 to 20 rounds.

Following the new trend of side-channel attacks, Handschuh and Heys [3] analyze RC5 with respect to timing attacks and show that the existence of data-dependent rotations in RC5 allows to recover the secret key with a few thousands of timing measurements only. This suggests RC5 should be implemented in a time constant way.

Recommended Reading

1. Biryukov A, Kushilevitz E (1998) Improved cryptanalysis of RC5. In: Nyberg K (ed) Advances in cryptology – EUROCRYPT'98. Lecture notes in computer science, vol 1403. Springer, Berlin, pp 85–99
2. Borst J, Preneel B, Vandewalle J (1999) Linear cryptanalysis of RC5 and RC6. In: Knudsen LR (ed) Fast software encryption – sixth international workshop. Lecture notes in computer science, vol 1636. Springer, Berlin, pp 16–30
3. Handschuh H, Heys H (1998) A timing attack on RC5. In: Tavares SE, Meijer H (eds) Selected areas in cryptography – SAC'98. Lecture notes in computer science, vol 1556. Springer, Berlin, pp 306–318
4. Kaliski BS, Yin YL (1995) On differential and linear cryptanalysis of the RC5 encryption algorithm. In: Coppersmith D (ed) Advances in cryptology – CRYPTO'95. Lecture notes in computer science, vol 963. Springer, Berlin, pp 171–184
5. Knudsen LR, Meier W (1996) Improved differential attacks on RC5. In: Koblitz N (ed) Advances in cryptology – CRYPTO'96. Lecture notes in computer science, vol 1109. Springer, Berlin, pp 216–228
6. Rivest R (1995) The RC5 encryption algorithm. In: Preneel B (ed) Fast software encryption – second international workshop. Lecture notes in computer science, vol 1008. Springer, Berlin, pp 86–96

RC6

Helena Handschuh
Computer Security and Industrial Cryptography Research Group, Katholieke Universiteit Leuven, Leuven - Heverlee, Belgium

Synonyms

Rivest cipher 6; Ron's code 6

Related Concepts

►AES Candidate; ►Block Ciphers; ►Symmetric Cryptography

Definition

RC6 is an iterative secret-key block cipher designed by Rivest, Robshaw, Sidney, and Yin [6] in 1998. It has variable parameters such as the key size, the block size, and the number of rounds. A particular (parameterized) RC6 encryption algorithm is designated as RC6 (w, r, b), where w is the word size (one block is made of four words), r is the number of rounds, and b is the number of bytes for the secret key.

Background

In 1997, the American National Institute for Standards and Technology (NIST) launched a competition for an Advanced Encryption Standard (AES) to replace their aging Data Encryption Standard (DES). RC6 is one of the five finalists of the contest, and its design is inspired and derived from its predecessor RC5. It was also submitted to the European NESSIE project and to the Japanese CRYPTREC project.

Theory

The three "nominal" choices for the algorithm as submitted to the American {Advanced Encryption Standard} (Rijndael/AES) contest and to the European NESSIE contest are

RC6 (32, 20, 16), RC6 (32, 20, 24), and RC6 (32, 20, 32). All three versions have a 128-bit block size, 20 rounds, and only differ in the key-size which is respectively 128, 196, and 256 bits long. The secret key is first expanded into an array of $2r+4$ secret w-bit words S_i according to the key scheduling algorithm. Let (A, B, C, D) denote the four w-bit words of the plaintext. Note that a w-bit word is equivalently viewed as an integer modulo 2^w (▸Modular Arithmetic). Then the encryption algorithm is given by:

```
$B \leftarrow B + S_0 \mod 2^w$
$D \leftarrow D + S_1 \mod 2^w$
$for\ i = 1\ to\ r\ do$
    $A \leftarrow ((A \oplus f(B)) \ll f(D)) + S_{2i} \mod 2^w$
    $C \leftarrow ((C \oplus f(D)) \ll f(B)) + S_{2i+1} \mod 2^w$
    $(A, B, C, D) \leftarrow (B, C, D, A)$
$A \leftarrow A + S_{2r+2} \mod 2^w$
$C \leftarrow C + S_{2r+3} \mod 2^w$
```

where "$\oplus$" represents bit-wise exclusive-or, "$X << Y$" is the (data-dependent) rotation of X to the left by the $\log_2 w$ least significant bits of Y, and the function f plays the role of a pseudo-random generator defined by:

$$f(x) = (x(2x+1) \mod 2^w) \ll \log_2 w.$$

The ciphertext is (A, B, C, D) and each round r involves exactly two subkeys S_i.

RC6 can be viewed as two parallel interleaved applications of RC5 with an extra multiplication operation.

Applications

RC6 overcomes certain weaknesses of its predecessor RC5 by introducing fixed rotations as well as a quadratic function to determine the data-dependent rotations. Contini et al. [1], on the one hand, and Iwata and Kurosawa [3], on the other hand, both give a comprehensive analysis of the contribution of these features to the security of RC6. The best statistical attack on RC6 by Gilbert et al. [2] breaks RC6 (32, 14, 16) and the best correlation attack by Knudsen and Meier [4] further enables a distinguisher and a key-recovery attack on RC6 (32, 15, 32). For a fraction of the keys called weak keys, RC6 is vulnerable to a multiple linear attack up to 18 rounds [7].

The more recent cryptanalysis results [5] mainly apply on reduced-round versions without whitening keys (called RC6W), i.e., versions without the first-round and/or last-round key addition and suggest that it remains difficult to attack more than 17 or 18 rounds of RC6W. Further results were obtained on several variants of RC6 such as ERC6 and MRC6, but the original nominal 20 round version of RC6 is still considered unbroken so far.

Recommended Reading

1. Contini S, Rivest RL, Robshaw MJB, Yin YL (1999) Improved analysis of some simplified variants of RC6. In: Knudsen LR (ed) Fast software encryption – seventh international workshop. Lecture notes in computer science, vol 1636. Springer, Berlin, pp 1–15
2. Gilbert H, Handschuh H, Joux A, Vaudenay S (2000) A statistical attack on RC6. In: Schneier B (ed) Fast software encryption – seventh international workshop. Lecture notes in computer science, vol 1978. Springer, Berlin, pp 64–74
3. Iwata T, Kurosawa K (2000) On the pseudorandomness of the AES finalists – RC6 and serpent. In: Schneier B (ed) Fast software encryption – seventh international workshop. Lecture notes in computer science, vol 1978. Springer, Berlin, pp 231–243
4. Knudsen LR, Meier W (2000) Correlations in RC6 with a reduced number of rounds. In: Schneier B (ed) Fast software encryption – seventh international workshop. Lecture notes in computer science, vol 1978. Springer, Berlin, pp 94–108
5. Miyaji A, Nonaka M (2002) Cryptanalysis of the reduced-round RC6. In: Deng R, Qing S, Bao F, Zhou J (eds) ICICS'2002. Lecture notes in computer science, vol 2513. Springer, Berlin, pp 480–494
6. Rivest RL, Robshaw MJB, Sidney R, Yin YL (1998) The RC6 block cipher. AES – The First Advanced Encryption Standard Candidate Conference, Conference Proceedings, NIST, 1998
7. Shimoyama T, Takenaka M, Koshiba T (2002) Multiple linear cryptanalysis of a reduced round RC6. In: Dalmen J, Rijmen V (eds) Fast software encryption – ninth international workshop. Lecture notes in computer science, vol 2365. Springer, Berlin, pp 76–88

Recipient Anonymity

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

Definition

Recipient ▸anonymity is achieved in a messaging system if an eavesdropper picks up messages from the communication line of a sender can – after some time of monitoring the network – not tell with better probability than pure guessing who has eventually received the messages. During the attack, the ▸eavesdropper may listen on all communication lines of the network including those that connect the potential senders to the network, he may 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. The ▸anonymity set for any particular message attacked by the eavesdropper is the set of all network participants that will have received a message within a certain time window after the attacked message was sent. This time window of course depends on latency characteristics and node configurations of the network itself.

Recipient anonymity against computationally unrestricted attackers can be achieved by broadcast, e.g., ►DC-Network [1, 2], by ►Mix-Network [2], or by anonymous information retrieval [3]. Note that recipient anonymity is weaker than *recipient unobservability*, where the attacker cannot even determine whether or not a participant has received a (meaningful) message. Recipient unobservability can be achieved with either of the above techniques by adding dummy traffic.

Recommended Reading

1. Chaum D (1985) Security without identification: transaction systems to make big brother obsolete. Commun ACM 28(10):1030–1044
2. Waidner M (1990) Unconditional sender and recipient untraceability in spite of active attacks. In: Quisquater J-J, Vandewalle J (eds) Advances in cryptography – EUROCRYPT'89. Lecture notes in computer science, vol 434. Springer, Berlin, pp 302–319
3. Chaum D (1981) Untraceable electronic mail, return addresses, and digital pseudonyms. Commun ACM 24(2):84–88
4. Cooper DA, Birman KP (1995) The design and implementation of a private message service for mobile computers. Wirel Netw 1:297–309

Record Linkage

►Data Linkage

Recursive Revoke

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

Synonyms

Cascade revoke

Related Concepts

►Grant Option; ►SQL Access Control Model

Definition

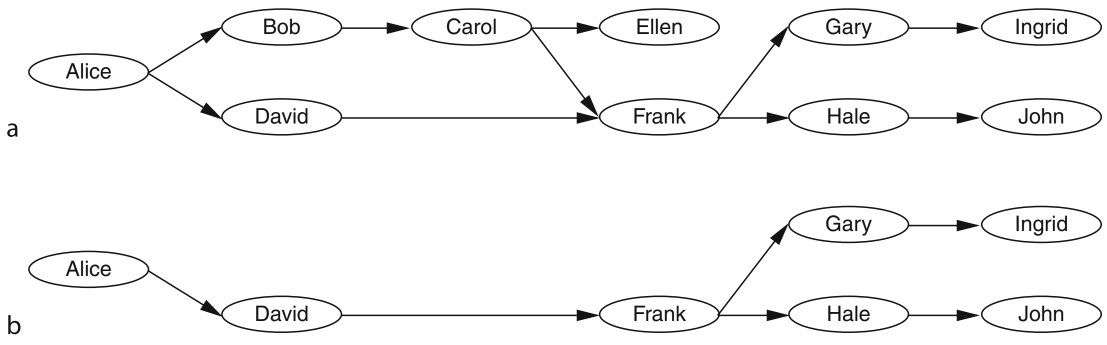
Recursive revocation demands that, when an administrative authorization is revoked, all the authorizations dependent on the revoked authorization be also revoked, iteratively applying the process for each authorization being revoked.

Theory

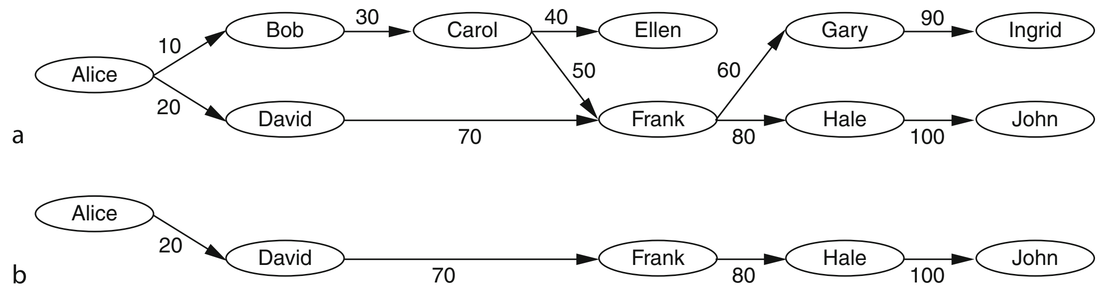
When administrative privileges can be delegated, if an administrative authorization is revoked, the problem arises of dealing with the authorizations specified by the users from whom the administrative privilege is being revoked. For instance, suppose that Alice gives Bob the authorization to read relation `Employee` with the ►grant option privilege, meaning that Bob can give this authorization to others. Suppose then that Bob grants the authorization to Carol, and that subsequently Alice revokes the authorization from Bob. The question now is: what should happen to the authorization that Carol has received? If Carol's authorization depended on the authorization being revoked from Bob, Carol's authorization would remain dangling and therefore it should also be removed. This would trigger the problem of the authorizations granted in turn by Carol, and recursively of other authorizations dependent on them, which should then be recursively deleted.

In the current SQL standard [2] recursive revocation can be requested by specifying option CASCADE when revoking privileges. With cascade revocation, when the user being revoked a privilege does not have anymore the administrative right (GRANT OPTION) for the privilege, all the authorizations she granted are automatically deleted. This control is also applied for each authorization deleted that had the grant option, causing revocation to recursively propagate if the user of the authorization does not have anymore the grant option for the privilege. To illustrate, consider the authorizations resulting from the grant operations (for a privilege, with the grant option) in Fig. 1a. Suppose that Alice revokes the authorization from Bob with cascade. Since Bob does not have anymore the grant option on the privilege, the authorization granted by Bob to Carol is also revoked. For the same reason, the authorizations granted by Carol to Ellen and Frank are revoked. Recursive revocation stops at Frank (the authorizations he granted are not revoked), in virtue of the authorization for the privilege with the grant option that Frank received from David. Figure 1b illustrates the authorization state resulting after the revocation.

While the common accepted interpretation of recursive revocation is SQL cascade revocation, it is worth noting that there is not a "unique" way to define which authorizations are dependent on another, and would then need to be recursively revoked since they would remain dangling if the other is deleted. For an example of alternative approach to recursive revocation, it is useful to



Recursive Revoke. Fig. 1 Grant operations (**a**) and the result of a `REVOKE` statement (**b**)



Recursive Revoke. Fig. 2 Example of the original System R, time-based cascade revocation

look at the history of System R [1]. The original System R revocation policy (*time-based cascade* revocation) adopted the following semantics: when a user is revoked the grant option on an access, all the authorizations that she granted and could not have been granted had the revoked authorization not been present, should also be (recursively) deleted. The revocation is recursive since it may, in turn, cause other authorizations to be deleted. More precisely, let $AUTH$ be the initial authorization state and $G_1, \ldots, G_n$ be a sequence of grant requests (history) that produced authorization state $AUTH'$. The revocation of a grant G_k should result in authorization state $AUTH''$ as if G_k had never been granted, that is, resulting from history $G_1, \ldots, G_{k-1}, G_{k+1}, \ldots, G_n$. Enforcement of this revocation semantics requires to keep track of (1) who granted which authorization and (2) the time at which the authorization was granted. To illustrate, assume that in the example mentioned above the time at which the authorizations were granted be as in Fig. 2a. The label associated with the arc states the time at which the authorization was granted (like before we assume all authorizations to be with the grant option). With time-based recursive revocation, when Alice revokes the authorization from Bob, the authorization from Carol will also be deleted (since Bob does not have anymore any authorization for the privilege with the grant option). Similarly, the authorization granted by Carol to Ellen and Frank will be revoked. Unlike before, revocation does not stop at Frank, since also the authorizations granted by Frank to Gary and by Gary to Ingrid need to be deleted. In fact, had the authorization that Frank received from David not existed, the grant request of Frank to give the authorization to Gary, and those of Gary to give it to Ingrid, would have been denied. Figure 2b illustrates the authorization state resulting after the revocation.

Although the time-based cascade revocation has a clean semantics, it may not be desirable in several scenarios, and hence the simpler approach to recursive revocation that was later adopted in SQL.

Note that recursive revocation needs to be applied carefully. In fact, deleting all authorizations granted in virtue of an authorization that is being revoked is not always wanted. In many organizations, the authorizations that users possess are related to their particular tasks or functions within the organization. Suppose there is a change in the task or function of a user (say, because of a job promotion). This change may imply a change in the responsibilities of the user and therefore in her privileges. New authorizations will be granted to the user and some of her previous authorizations will be revoked. Applying a recursive revocation will result in the undesirable effect of deleting all authorizations the revokee granted and, recursively, all the authorizations granted through them, which then will need to be reissued. Moreover, all application programs depending on the revoked authorizations will be invalidated. With the non-cascade option the system rejects the revoke operation if its enforcement would entail deletion of other authorizations beside the one for which revocation is requested.

Recommended Reading

1. Griffiths PP, Wade BW (Sept 1976) An authorization mechanism for a relational database system. ACM Trans Database Syst (TODS) 1(3):242–255
2. Database Language SQL (2008) ISO International Standard, ISO/IEC 9075–*:2008

Reed–Muller Codes

Pascale Charpin
INRIA, Rocquencourt, Le Chesnay Cedex, France

Synonyms

Boolean functions

Related Concepts

►Boolean Functions; ►Extended Cyclic Code; ►Information Theory; ►Symmetric Cryptography

Definition

The Reed–Muller code of length 2^m and order r is the linear binary code whose codewords can be identified to Boolean functions of m variables and of algebraic degree less than or equal to r.

Background

Coding theory

Theory

It is well known that any property of Reed–Muller codes is a property of ►Boolean functions. Reed–Muller codes provide a natural way to quantify the *degree*, the nonlinearity, the ►correlation-immunity, or the ►propagation criterion of a Boolean function [1]. On the other hand, Reed–Muller codes are an important class of error-correcting codes, in particular they can be viewed as extended ►cyclic codes. They play a crucial role in the study of important families of cryptographic mappings, such as permutations on ►finite fields. Here, the multivariable definition of Reed–Muller codes is presented. More on Reed–Muller codes can be found in [2].

Definition 1 *Define $\{\mathbb{F}_2^m, +\}$ as an ordered vector space:*

$$\mathbb{F}_2^m = \{\mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_{2^m-1}\}, \tag{1}$$

where $\mathbf{v}_i$ is an m-dimensional binary vector (often one takes $\mathbf{v}_i$ as the binary representation of integer i). The Reed–Muller code of length 2^m and order r, $0 \le r \le m$, denoted by $\mathcal{R}(r,m)$, is the binary code of length 2^m consisting of all codewords $(f(\mathbf{v}_0), f(]\mathbf{v}_1), \ldots, f(\mathbf{v}_{2^m-1}))$ *where f is any Boolean function of m variables whose algebraic degree is less than or equal to r.*

Note that $\mathcal{R}(0,m)$ corresponds to the set of constant functions and that $\mathcal{R}(1,m)$ corresponds to the set of functions that are affine or constant.

The r-th order Reed–Muller code is usually constructed by using as basis the set of *monomials* of degree at most r. One can interpret $\mathbb{F}_2^m$ as an $m \times 2^m$ binary matrix, the ith row of which is the sequence of values of variable $\mathbf{x}_i$ evaluated for the successive vectors $\mathbf{v}_i$. Thus, $\mathbf{x}_i$ is in fact a Boolean function over $\mathbb{F}_2^m$: for each j, $\mathbf{x}_i(\mathbf{v}_j)$ equals the ith symbol of vector $\mathbf{v}_j$. This function produces a unique codeword whose symbols are labeled with respect to (1). For the sake of simplicity, $\mathbf{x}_i(\mathbf{v}_j)$ is denoted by $x_{i,j}$.

Now, any monomial $\prod_{i \in I} \mathbf{x}_i$ is, in the same way, a Boolean function g satisfying for $0 \le j \le 2^m - 1$

$$g(\mathbf{v}_j) = \begin{cases} 1, & \text{if } x_{i,j} = 1 \text{ for all } i \in I, \\ 0, & \text{otherwise.} \end{cases}$$

In this way, a basis for $R(r,m)$ can be obtained. An explicit construction for $m = 4$ is presented in the example below.

Proposition 1 *The set of monomials*

$$\mathbf{x}_1^{e_1}\mathbf{x}_2^{e_2}\cdots\mathbf{x}_m^{e_m}, \quad e_i \in \{0,1\}, \qquad \sum_{i=1}^{m} e_i \le r,$$

is a basis of code $\mathcal{R}(r,m)$.

Example : Construction of a generator matrix $\mathcal{G}_r$ for each code $\mathcal{R}(r,4)$, $0 \le r \le 4$. Matrix $\mathcal{G}_r$ is obtained by computing the codewords produced by the monomials of degree less than or equal to r, using Proposition 1. Observe in Table 1 that $\mathcal{G}_1$ is a 5×16 matrix, $\mathcal{G}_2$ is a 11×16 matrix, etc. Note also that the lines of $\mathcal{G}_1$ are the all-one vector and the lines of matrix $\mathbb{F}_2^4$ (with the given fixed order).

Reed–Muller Codes. Table 1 Generator matrices of the codes $\mathcal{R}(r,4)$

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	$\mathcal{G}_0$
$\mathbf{x}_4$	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	
$\mathbf{x}_3$	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	
$\mathbf{x}_2$	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	
$\mathbf{x}_1$	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	$\mathcal{G}_1$
$\mathbf{x}_3\mathbf{x}_4$	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	
$\mathbf{x}_2\mathbf{x}_4$	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	1	
$\mathbf{x}_1\mathbf{x}_4$	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	1	
$\mathbf{x}_2\mathbf{x}_3$	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1	
$\mathbf{x}_1\mathbf{x}_3$	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0	1	
$\mathbf{x}_1\mathbf{x}_2$	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	$\mathcal{G}_2$
$\mathbf{x}_2\mathbf{x}_3\mathbf{x}_4$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	
$\mathbf{x}_1\mathbf{x}_3\mathbf{x}_4$	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	
$\mathbf{x}_1\mathbf{x}_2\mathbf{x}_4$	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	
$\mathbf{x}_1\mathbf{x}_2\mathbf{x}_3$	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	$\mathcal{G}_3$
$\mathbf{x}_1\mathbf{x}_2\mathbf{x}_3\mathbf{x}_4$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	$\mathcal{G}_4$

Clearly, $\mathcal{R}(r,m)$ is a linear subspace of the 2^m-dimensional binary vector space. One can easily verify that its dimension is

$$\kappa_r = \dim \mathcal{R}(r,m) = 1 + m + \ldots + \binom{m}{r}. \tag{2}$$

It can also be shown (see [2]) that different codewords in $\mathcal{R}(r,m)$ differ in at least 2^{m-r} coordinates. One says that $\mathcal{R}(r,m)$ has *minimum distance* 2^{m-r} (►Boolean functions). This proves:

Proposition 2 *The r-th order Reed–Muller code $\mathcal{R}(r,m)$, $0 \le r \le m$, has length $n = 2^m$, dimension κ_r given by (2), and minimum distance 2^{m-r}.*

Recommended Reading

1. Massey JL (1995) Some applications of coding theory in cryptography. In: Farell PG (ed) Codes and ciphers: cryptography and coding IV. Formara, Essex, pp 33–47
2. Pless VS, Huffman WC, Brualdi RA (1998) An introduction to algebraic codes. Handbook of coding theory, part 1: algebraic coding, chapter 1. Elsevier, Amsterdam

Reference Monitor

Trent Jaeger
Systems and Internet Infrastructure Security Lab, Pennsylvania State University, University Park, PA, USA

Related Concepts

►Access Control from an OS Security Perspective; ►Access Control Policies, Models, and Mechanisms; ►Bell-LaPadula Confidentiality Model; ►Biba Integrity Model; ►Security Kernel

Definition

A *reference monitor concept* defines a set of design requirements on a *reference validation mechanism*, which enforces an access control policy over subjects' (e.g., processes and users) ability to perform operations (e.g., read and write) on objects (e.g., files and sockets) on a system.

- The reference validation mechanism must always be invoked (*complete mediation*).
- The reference validation mechanism must be tamperproof (*tamperproof*).
- The reference validation mechanism must be small enough to be subject to analysis and tests, the completeness of which can be assured (*verifiable*).

The claim is that a reference validation mechanism that satisfies the reference monitor concept will correctly enforce a system's access control policy, as it must be invoked to mediate all security-sensitive operations, must not be tampered, and has undergone complete analysis and testing to verify correctness.

Background

In 1972, James P. Anderson coordinated the Computer Security Technology Planning Study with a panel of ten industry, government, and academic security experts [2]. The goal of the panel was to determine the requirements for US government computer systems to execute securely in the presence of malicious users. Contemporary systems assumed that all threats originated from external attackers, but malicious users in control of processes running on the system may also violate the system's access control policy. For example, a malicious programmer may steal information accessed by processes running her program or a malicious user may steal other user's data stored on the same system.

To prevent unauthorized access by malicious users, the panel recommended that computing systems be designed in accordance with requirements embodied by the *reference monitor concept*. The reference monitor concept envisions that a system component, called a reference validation mechanism, will be responsible for enforcing the system's access control policy over user process operations. The reference monitor concept defines the requirements for implementing such a mechanism in a manner that ensures that malicious users cannot circumvent policy enforcement.

Theory

The claim is that by implementing a reference validation mechanism in accordance with the reference monitor concept all accesses by user processes will adhere to an access control policy enforced by the mechanism. This claim is based on successful implementation of the three design requirements of the definition above.

First, the *complete mediation requirement* specifies that the reference validation mechanism mediates all security-sensitive operations by user processes. Complete mediation enables the reference validation mechanism to authorize each security-sensitive operation against the access control policy. Since only security-sensitive operations can violate the access control policy, such mediation ensures that user processes can only perform operations authorized by the access control policy.

Second, the *tamperproof requirement* specifies that the reference validation mechanism cannot be modified by user processes. This prevents a malicious user from modifying the behavior of the reference validation (mechanism), e.g., to approve operations that are not allowed by the access control policy. In practice, the tamperproof requirement also covers the access control policy itself. This prevents a malicious user from gaining unauthorized access by modifying the access control policy enforced by the reference validation mechanism.

Third, the *verifiable requirement* specifies that the reference validation mechanism be small enough to enable practical verification of correctness. A reference validation mechanism is correct if it generates the correct access control query, processes that query correctly against the access control policy, and correctly implements the resultant decision (allow or deny). While not strictly implied by the requirement, it is also desirable to determine whether the access control policy is correctly specified relative to some goal.

Applications

The Multics operating system [14] was the first one designed with comprehensive security enforcement, aiming to protect the secrecy of data and the integrity of the operating system and trusted software. However, as the Multics project matured, it became clear that the Multics system was complex, and this hindered the developers ability to determine whether the proposed security guarantees were correctly enforced [13].

The development of the reference monitor concept aimed to remedy this problem by defining the requirements for a correct and unbypassable reference validation mechanism. Subsequently, several projects emerged to demonstrate the efficacy of building reference validation mechanisms based on these design requirements. These types of operating systems were called *security kernels*, as they each contained a small, core component that included a reference validation mechanism designed to satisfy the reference monitor concept [1]. Examples of commercial security kernels were Scomp [4] and gemini multiprocessing secure operating system (GEMSOS) [12].

Since that time, the reference monitor concept has been considered a foundation of secure system design. This is reflected in the US Government's criteria for building secure systems, the Trusted Computer System Evaluation Criteria [17] (in particular, the "Orange" book), where the reference monitor is used as a motivation for the choice of its security evaluation classes. Gasser's seminal book [5], "Building a Secure Computer System," also based its design methodology on the reference monitor concept. This book motivated a variety of researchers and companies to leverage microkernel and hypervisor systems for security enforcement, as their small size made them more amenable to verification. Finally, Irvine suggested that the reference monitor concept be used as a guiding principle for computer security education [6].

Despite this convergence on the reference monitor concept in the security community [3, 9, 11], commercial operating system designs were largely unaffected. Trusted Solaris [15] was an exception, as the Solaris system was forked in the late 1980s to deploy an operating system for enforcing multilevel security. However, this version of Solaris remained separate from mainstream Solaris until 2006. It was not until the early 2000s, when the fundamental security problems in commercial systems were finally accepted [10], that a reference validation mechanism aiming for the reference monitor concept was developed for a mainstream, commercial operating system. The Linux security modules (LSM) framework was implemented for Linux 2.6 [18], enabling the support of multiple reference validation mechanism. That is, the LSM framework provides complete mediation, whereas the choice of module (and supporting system services) determines how tamperproofing and verification are achieved. A similar design has been applied to TrustedBSD, Mac OS X, the Xen hypervisor, and some user-space programs, most notably X Windows.

Open Problems

The main open problems in meeting the requirements of the reference monitor concept involve verifying those requirements on implementations of reference validation mechanisms [7]. Each requirement has its own unique challenges. Complete mediation requires that all security-sensitive operations are identified, but often these operations are not precisely defined. Tools have been built to verify complete mediation for the LSM framework [8, 16, 19], and several bugs were found (since fixed). For tamperproofing, the problem is that most systems have a trusted computing base that is too large to determine whether tampering is prevented. In many systems, several user-level processes are trusted with the authority to modify the kernel (e.g., install modules), but many of these processes themselves are vulnerable to tampering. However, verification is the most difficult of the requirements to satisfy, as designing a general algorithm to prove that an arbitrary program behaves correctly is tantamount to solving the Halting problem. While current algorithms can prove correctness properties of specific programs, the variety of reference validation code and the complexity of correctness properties preclude verification for all but the smallest, most specialized systems.

Recommended Reading

1. Ames SA, Gasser M, Schell RR (1983) Security kernel design and implementation: an introduction. IEEE Comp 16(7): 14–22
2. Anderson JP (1972) Computer security technology planning study. Technical report ESD-TR-73-51, http://seclab.cs.ucdavis.edu/projects/history/, The Mitre Corporation, Air Force Electronic Systems Division, Hanscom AFB, Badford. Volumes I and II
3. Branstad M, Tajalli H, Mayer FL, Dalva D (1989) Access mediation in a message passing kernel. In: Proceedings of the 1989 IEEE Symposium on Security and Privacy, Oakland, CA
4. Fraim LJ (1983) SCOMP: a solution to the multilevel security problem. IEEE Comp 16(7):26–34
5. Gasser M (1988) Building a secure computer system. Van Nostrand Reinhold, New York. http://cs.unomaha.edu/~stanw/gasserbook.pdf
6. Irvine C (1999) The reference monitor concept as a unifying principle in computer security education. In: Proceedings of the 1st world conference on information systems security education, Kista, Sweden, June 1999
7. Jaeger T (2008) Operating system security. Morgan & Claypool, San Rafael, CA
8. Jaeger T, Edwards A, Zhang X (2004) Consistency analysis of authorization hook placement in the Linux security modules framework. ACM Trans Inform Syst Sec (TISSEC) 7(2): 175–205
9. Karger PA, Zurko ME, Bonin DW, Mason AH, Kahn CE (1991) A retrospective on the VAX VMM security kernel. IEEE Trans Softw Eng 17(11):1147–1165
10. Loscocco PA, Smalley SD, Muckelbauer PA, Taylor RC, Turner SJ, Farrell JF (1998) The inevitability of failure: the awed assumption of security in modern computing environments. In: Proceedings of the 21st National Information Systems Security Conference, Arlington, VA, pp 303–314, October 1998
11. Minear SE (1995) Providing policy control over object operations in a Mach-based system. In: Proceedings of the 5th USENIX Security Symposium, Salt Lake City, UT, pp 141–156
12. Schell R, Tao T, Heckman M (1985) Designing the GEMSOS security kernel for security and performance. In: Proceedings of the National Computer Security Conference, Baltimore, MD
13. Schroeder MD (1975) Engineering a security kernel for Multics. In: Proceedings of the Fifth ACM Symposium on Operating Systems Principles, Austin, TX, pp 25–32
14. Schroeder MD, Clark DD, Saltzer JH, Wells D (1978) Final report of the MULTICS kernel design project. Technical report MIT-LCS-TR-196, MIT, March 1978
15. Sun Microsystems. Trusted Solaris 8 Operating System. http://www.sun.com/software/solaris/trustedsolaris/, February 2006
16. Tan L, Zhang X, Ma X, Xiong W, Zhou Y (2008) AutoISES: automatically inferring security specifications and detecting violations. In: Proceedings of the 17th USENIX Security Symposium, USENIX Association, San Jose, CA, pp 379–394
17. Trusted Computer System Evaluation Criteria (Orange Book). Technical report DoD 5200.28-STD, U.S. Department of Defense, December 1985
18. Wright C, Cowan C, Smalley S, Morris J, Kroah-Hartman G (2002) Linux security modules: general security support for the Linux kernel. In: Proceedings of the 11th USENIX Security Symposium, San Francisco, CA, pp 17–31, August 2002
19. Zhang X, Edwards A, Jaeger T (2002) Using CQUAL for static analysis of authorization hook placement. In: Proceedings of the 11th USENIX Security Symposium, San Francisco, CA, pp 33–48, August 2002

Related Key Attack

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

Related Concepts

►Block Ciphers; ►IDEA; ►Rijndael; ►Stream Cipher; ►Triple DES

Definition

The idea of the attack is that the attacker knows (or chooses) a relation between several keys (up to 256 in some recent attacks) and is given access to encryption functions with such related keys. The goal of the attacker is to find the keys themselves.

Background

The first attacks of this type were developed independently by Biham [1] and Knudsen [2], and the notion of a *related key attack* was defined by Biham [1].

Theory

The relation between the keys can be an arbitrary bijective function *R* (or even a family of such functions) chosen (or known) in advance by the adversary [3]. In the simplest form of this attack, this relation is just a XOR with a constant: $K_2 = K_1 + C$, where the constant *C* is chosen by the adversary. This type of relation allows the adversary to trace the propagation of XOR differences induced by the key difference *C* through the key schedule of the cipher. However, more complex forms of this attack allow other (possibly nonlinear) relations between the keys. For example, [1] uses rotational relations $K_2 = ROL(K_1)$ and the recent attack on AES-256 [4] uses XOR relations between the subkeys $K_2 = F^{-1}(F(K_1) + C) = R_C(K_1)$.

Applications

A line of ciphers have been shown to have weaknesses in this attack scenario [5, 6], namely, IDEA, GOST, SAFER, Triple-DES, CAST, DES-X, TEA, etc. Widely used ciphers like RC4 [7], Kasumi [8], AES-192, and AES-256 [4, 9] have been broken in this attack scenario.

The scenario of the attack is very powerful in terms of the attacker's capabilities and thus somewhat unrealistic in practice. However, cryptographers usually try to design primitives which can be automatically used without further analysis in the widest possible set of applications, protocols, or modes of operation. Thus, security against such attacks is always one of the design goals. For example, block ciphers susceptible to related-key differential attacks may lead to insecure compression functions, if they are instantiated by a Davies-Meyer construction (e.g., [9]).

A cryptanalytic attack called slide attack can be viewed as a variant of a related key attack, in which a relation of the key with itself is exploited. Slide attacks are known plaintext or chosen plaintext attacks and thus are more practical than related key attacks since they do not require the attacker to know relations between different encryption keys.

Recommended Reading

1. Biham E (1994) New types of cryptanalytic attacks using related keys. Journal of Cryptol 4:229–246. In: Helleseth T (ed) An earlier version appeared in the proceedings of *EUROCRYPT'93*. Lecture Notes in Computer Science, vol 765
2. Knudsen LR (1993) Cryptanalysis of LOKI91. In: Seberry J, Zheng Y (eds) Advances in cryptography – ASIACRYPT'92. Lecture notes in computer science, vol 718. Springer, Berlin, pp 22–35
3. Belare M, Khono T (2003) "A theoretical treatment of related-key attacks", RKA-PRPs, RKA-PRFs, and Applications. In: Biham E (ed) Advances in cryptology – EUROCRYPT 2003. Lecture notes in computer science, vol 2656. Springer, Berlin, pp 491–506
4. Biryukov A, Khovratovich D (2009) Related-Key Attack on the Full AES-192 and AES-256. In: Halevi S (ed) Advances in cryptology – ASIACRYPT'09. Lecture notes in computer science, vol 5912. Springer, Berlin, pp 1–18
5. Kelsey J, Schneier B, Wagner D (1996) Key-schedule cryptanalysis of IDEA, G-DES, GOST, SAFER, and Triple-DES. In: Koblitz N (ed) Advances in cryptology – CRYPTO'96. Lecture notes in computer science, vol 1109. Springer, Berlin, pp 237–251
6. Kelsey J, Schneier B, Wagner D (1997) Related-key cryptanalysis of 3-WAY, Biham-DES, CAST, DES-X, NewDES, RC2, TEA. In: Han Y, Okamato T, Qing S (eds) International conference on information and communications security – ICICS'97. Lecture notes in computer science, vol 1334. Springer, Berlin
7. Fluhrer SR, Mantin I, Shamir A (2001) Weaknesses in the key scheduling algorithm of RC4. Selected areas in cryptography: pp 1–24
8. Biham E, Dunkelman O, Keller N (2005) A related-key rectangle attack on the full KASUMI. ASIACRYPT, pp 443–461
9. Biryukov A, Khovratovich D, Nikolic I (2009) Distinguisher and related-key attack on the full AES-256. CRYPTO, pp 231–249

Relationship Anonymity

►Communication Channel Anonymity

Relatively Prime

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

R

Synonyms

Coprime

Related Concepts

►Greatest Common Divisor; ►Prime Number

Definition

Two integers *n* and *x* are *relatively prime* if they have no common integer factors other than +1 and −1, i.e., their ►greatest common divisor is 1.

Relay Attack

Yvo Desmedt
Department of Computer Science, University College London, London, UK

Definition

An attack in which resources of a third party are used.

Theory

In a *relay attack*, a party, let us say Eve, will use the resources of a second party in an unauthorized way. A typical example is related to e-mail. Suppose Eve wants to send spam e-mail to lots of users, but does not have the resources (e.g., bandwidth). She will try to use Alice's machine to have it send all this e-mail. This may result in a denial of service against Alice's machine. Old mail servers allow mail to be relayed.

Remote Attestation

►Phenotyping

Replay Attack

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

Related Concepts

►Impersonation Attack

Definition

A *replay attack* is an attack in which the adversary records a communication session and replays the entire session, or some portion of the session, at a later point in time. The replayed message(s) may be sent to the same verifier as the one that participated in the original session, or to a different verifier. The goal of the replay attack may be impersonation (►Impersonation Attack), or it may be some other deception (e.g., a successful protocol exchange to transfer money from A's account to B's account may be replayed by B in an attempt to transfer more money than A had intended) [1, 2].

Recommended Reading

1. Menezes A, van Oorschot P, Vanstone S (1997) Handbook of applied cryptography. CRC, Boca Raton, FL
2. Schneier B (1996) Applied cryptography: protcols, algorithms, and source code in C, 2nd edn. Wiley, New York

Residue Arithmetic

►Modular Arithmetic

Resistance to the Standard Algebraic Attack

►Algebraic Immunity of Boolean Functions

Response

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

Related Concepts

►Challenge–Response Protocol

Definition

In a cryptographic identification scheme, a *response* is the answer by a claimant to a question posed by a verifier in a ►Challenge–Response Protocol. More generally, a response is any answer to a question or statement made by another party.

Resynchronization Attack

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

Related Concepts

►Stream Cipher

Definition

►Synchronous stream ciphers require some procedure for resynchronizing in the case of synchronization loss. This opens doors to new attack scenarios. A typical ►stream cipher encrypts the stream in fixed data blocks, called *frames* (or packets), by keeping the same secret ►key for all the frames but mixing the new *initial value (IV)* or the *frame-counter* for each frame (e.g., the ►A5/1 cipher). This allows for easy synchronization as well as for the late entry mechanism in the case of multiparty communication. On the one hand, such mode of operation produces only short streams for any fixed state which reduces the chances of some attacks, but on the other hand, it may open doors to new analysis techniques which will attack the resynchronization mechanism itself. Depending on the way IV and the key are loaded and mixed into the *state* of the stream cipher, the scheme may be susceptible to the standard arsenal of attacks developed for ►block ciphers such as differential, linear, slide, or other attacks.

Applications

Resynchronization attacks can be used in *known* or *chosen IV* scenarios. A typical resynchronization attack on stream ciphers is given in [3]. For more results on the subject see [1, 2, 4–6].

Recommended Reading

1. Armknecth F, Lano J, Preneel B (2004) Extending the resynchronization attack. In: Proceedings of selected areas in CRYPTOGRAPHY 2004. Lecture notes in computer science, vol 3357. Springer, Berlin, pp 19–38
2. Biryukov A, Wagner D (1999) Slide attacks. In: Knudsen LR (ed) Proceedings of fast software encryption—FSE'99. Lecture notes in computer science, vol 1636. Springer, Berlin, pp 245–259
3. Borissov Y, Nikova S, Preneel B, Vandewalle J (2002) On a resynchronization weakness in a class of combiners with memory. In: Cimato S, Galdi C, Persiano G (eds) Third international conference, SCN 2002, Amalfi, Italy, 11–13 Sept. Revised papers, Lecture notes in computer science, vol 2576. Springer, Berlin, pp 164–173
4. Daemen J, Govaerts R, Vandewalle J (1993) Resynchronization weaknesses in synchronous stream ciphers. In: Helleseth T (ed) Advances in cryptology—EUROCRYPT'93 Lecture notes in computer science, vol 765. Springer, Berlin, pp 159–167
5. Ekdahl P, Johansson T (2003) Another attack on A5/1. IEEE Trans Inform Theory 49:1–7
6. Golic JD, Morgari G (2003) On resynchronization attack. In: Johansson T (ed) Fast software encryption, FSE 2003. Lecture notes in computer science, vol 2887. Springer, Berlin, pp 100–110

Reverse Engineering of Malware Emulators

Michalis Polychronakis
Department of Computer Science, Network Security Lab, Columbia University, New York, NY, USA

Synonyms

VM protection; Virtualized packer

Definition

Reverse engineering of malware emulators deals with the analysis of obfuscated malicious code that has been transformed into a custom instruction set, interpreted at runtime by an embedded emulator.

Background

Virtualization-based obfuscation is one of the most advanced and hard to reverse engineer code obfuscation techniques. It has been used in commercial software protection products such as Code Virtualizer, VMProtect, and Themida since the early 2000s, and consequently gained the attention of malware authors who always seek more sophisticated methods to evade malicious code analysis and detection systems.

Applications

Code obfuscation is used by programmers to conceal the actual structure of the original code and hinder tampering or reverse engineering attempts. Malware authors employ code obfuscation to evade antivirus scanners and obstruct automated or manual analysis of the malicious code.

Code obfuscation based on emulation, also known as virtual machine (VM) protection, is one of the most sophisticated anti-reverse engineering techniques. Using this technique, each copy of the original binary program, written for a native *source* instruction set architecture (ISA) such as x86, is transformed into a custom assembly-like language that conforms to a new, hypothetical *target* ISA. At runtime, an embedded virtual machine emulates the execution of the target ISA on the real machine. The new instruction set is generated at random, and thus the language specification of the target ISA remains unknown, rendering traditional code analysis techniques ineffective.

The analysis of a malware sample armored using VM protection requires the extraction of the semantics of the custom instruction set of the malicious code. These semantics are hard-coded into the VM that emulates the target

ISA, and thus, a mapping between the target and the source ISA can be extracted by reverse engineering the emulator that is embedded in the malware sample. However, this is a nontrivial task since the code of the VM itself is usually heavily obfuscated. Once such a mapping exists, the malicious code can be translated into its original source ISA, and then analyzed using existing tools and techniques.

Open Problems

Reverse engineering of malware emulators is a relatively new research area. Initial efforts relied on manual analysis of the template language used by some VM protection engines for the generation of the target ISA [1]. However, constructing a custom parser for each template is a tedious and time-consuming process, and has to be repeated for each instance of code generated by more advanced obfuscation engines that produce totally random VMs. The only existing approach for automatic reverse engineering of VM protection is effective only for a particular class of VMs that use interpretive emulation with a distinctive instruction decode and dispatch loop [2]. Generic automated reverse engineering of malware emulators that use more sophisticated types of emulation remains an open problem.

Recommended Reading

1. Rolles R (2009) Unpacking virtualization obfuscators. In: Proceedings of the 3rd USENIX workshop on offensive technologies (WOOT), Montreal, 10–14 Aug 2009
2. Sharif M, Lanzi A, Giffin J, Lee W (2009) Automatic reverse engineering of malware emulators. In: Proceedings of the 30th IEEE symposium on security and privacy, Oakland, 17–20 May 2009

Reverse Public Key Encryption

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

Related Concepts

►Public-Key Encryption

Definition

Reverse Public-Key Encryption is a low-bandwidth public-key encryption mode of operation. The construction turns a weak form of key privacy into message privacy as follows: let $\mathcal{E}$ be a public-key encryption algorithm. If the distributions $\mathcal{E}(pk_0, \bullet)$ and $\mathcal{E}(pk_1, \bullet)$ are indistinguishable for two public keys pk_0 and pk_1, then a message bit $b \in \{0,1\}$ can be embedded in the choice of pk_b. As the roles of the public-key and the plaintext are reversed, this mode is referred to as *Reverse Public-Key Encryption*.

Recommended Reading

1. Naccache D, Steinwandt R, Yung M (2009) Reverse public key encryption. BIOSIG 2009 proceedings, vol 155 of Lecture notes in informatics, GI, Springer, pp 155–169

RFID Security

Gildas Avoine
Université catholique de Louvain, Louvain-la-Neuve, Belgium

Related Concepts

►Contactless Smartcards; ►Passport Security; ►Relay Attack

Definition

RFID security is an active engineering and research domain that focuses on security of contactless integrated circuits and their applications.

Background

Radio Frequency Identification (RFID) allows to identify and authenticate objects or subjects without any physical nor optical contact, using transponders – integrated circuits including an antenna – queried by readers through a radio frequency channel. This technology is one of the most promising of this decade and is already widely used in applications such as access cards, public transportation, payment cards, biometric passports, and pet identification. This success is partly due to the steady decrease in both size and cost of the transponders, commonly called *tags*.

Theory

While RFID exists for several decades, RFID security only became recently a major concern in the information security community. Expertise in security of smartcards was already well established when RFID arrived under the spotlights in the 1990s, but the RFID technology raised new problems unexplored until then. The fact that most of the tags are batteryless and resource-constrainted leads to new challenges on lightweight cryptographic designs and implementations. Also, tags communicate wirelessly and usually without agreement or awareness of their holders, which modifies the threat model compared to smartcards with contacts.

Initially, researchers tried to distinguish the RFID technology from the contactless smartcard technology.

Nowadays, the trend is that RFID no longer targets identification only but covers contactless smartcards as well. The RFID security issues are commonly classified into three families that are described below: impersonation, privacy, and denial of service.

Impersonation

Many evolved applications require authentication and not only identification. This security objective does not refer to RFID environments only, but low-capability tags make this objective hardly reachable without degrading security. Impersonation includes "cloning" that consists in copying the public and private data of a given tag on another one. Another major concern is the relay attack [5, 8] that defeats any classical authentication protocol by relaying the low-layer communication between the parties involved in the protocol with the objective to cheat on the distance between them. Intensive research activities recently started on this topic.

Practical impersonation attacks hit the headlines for a few years. One may cite the attacks against the Texas Instruments' Digital Signature Transponder [3], the Keeloq ignition car system [2], the Mifare Classic card [10], among the most famous ones.

Privacy

RFID also raises controversial questions about privacy. Privacy advocates claim that RFID endangers individual liberties and consider it as a way of obtaining information about citizens without their consent [4, 6]. Among privacy, one may distinguish *information leakage* where the tag or the back-end reveals some private – possibly personal – information, from *illicit tracking* that consists in tracking a tag in different places or at different times without the agreement of its holder. Defining privacy models and designing privacy-friendly RFID solutions is still a challenging task which many research works focus on.

Denial of service

Denial of service, which is far less considered by the research community, may have awful consequences in large-scale or sensitive RFID applications, e.g., when tags are used to control a supply chain. A denial of service can be perpetrated using some electromagnetic means to noise the tag-reader channel, but it can also straightly target the back-end system or the tag. For example, it can exploit a bug in the reader firmware to crash the back-end system, or it can kill tags with an RFID-zapper [9] or the kill-command of the tag, when available [7].

Further Readings

There exists an abundant literature devoted to RFID security and privacy, with several hundred scientific articles published during the last 10 years. The RFID Security and Privacy Lounge [1] is a source of information that references most of the scientific articles published on the topic so far.

For practical handling, many open-source libraries for operating readers and tags are available on Internet. The website YobiWiki [11] references the most widely used ones, e.g., librfid, libnfc, rfidiot, and many others.

Recommended Reading

1. Avoine G (2004) RFID security and privacy lounge. http://www.avoine.net/rfid/. Accessed 6 Mar 2011
2. Biham E, Dunkelman O, Indesteege S, Keller N, Preneel B (2007) How to steal cars – A practical attack on keeLoq. http://www.cosic.esat.kuleuven.be/keeloq/. Accessed 6 Mar 2011
3. Bono S, Green M, Stubblefield A, Rubin A, Juels A, Szydlo M (2005) Analysis of the Texas instruments DST RFID. http://www.carthiefstoppers.com/About-RFIDs-and-the-Texas-Intruments-DST.html. Accessed 6 Mar 2011
4. CASPIAN. Consumers against supermarket privacy invasion and numbering. http://www.nocards.org/. Accessed 6 Mar 2011
5. Desmedt Y, Goutier C, Bengio S (1988) Special uses and abuses of the Fiat-Shamir passport protocol. In: Carl pomerance (ed) Advances in cryptology – CRYPTO'87, IACR Santa Barbara, CA. Lecture notes in computer science, vol 293. Springer, Heidelberg, pp 21–39
6. Electronic Privacy Information Center. Radio frequency identification (RFID) systems. http://epic.org/privacy/rfid/. Accessed 6 Mar 2011
7. EPC Global Inc. (2008) EPC radio-frequency identity protocols – Class-1 generation-2 UHF RFID – Protocol for communications at 860 MHz–960 MHz – Version 1.2.0. Technical report, EPC Global Inc.
8. Hancke G, Kuhn M (2005) An RFID distance bounding protocol. In: Conference on security and privacy for emerging areas in communication networks – SecureComm 2005, Athens, Greece
9. MiniMe T, Mahajivana C (2005) RFID Zapper. Computer chaos congress 2005. Berlin, Germany
10. NXP Semiconductors. Security of MIFARE Classic. http://www.mifare.net/technology/security/. Accessed 6 Mar 2011
11. Philippe Teuwen (2009) YobiWiki. http://wiki.yobi.be/wiki/RFID. Accessed 6 Mar 2011

Right-to-Left Exponentiation

Bodo Möller
Google Switzerland GmbH, Zurich, Switzerland

Related Concepts

►Exponentiation Algorithms

Definition
Right-to-left exponentiation methods read the exponent from the least significant bit (or other digit) up to the most significant one.

Background
Many exponentiation methods have two variants: one that examines exponents starting at the most significant digit and going down to the least significant one, that is, in *left-to-right* direction (assuming big-endian notation); and a related one that examines exponents in the opposite direction, that is, *right-to-left*. For specific methods, see the entries on ►binary exponentiation, ►2^k-ary exponentiation, and ►sliding window exponentiation.

Theory
There is a general duality between left-to-right and right-to-left exponentiation: this is explained by representing addition chains (►fixed-exponent exponentiation) as directed multi-graphs such that reversing all arcs turns left-to-right exponentiation into right-to-left exponentiation, or the other way around [1].

Recommended Reading
1. Knuth DE (1998) The art of computer programming, vol. 2: seminumerical algorithms, 3rd edn. Addison-Wesley, Reading, p 466, 639, 4.6.3–39

Rijndael

Joan Daemen[1], Vincent Rijmen[2]
[1]STMicroelectronics, Zaventem, Belgium
[2]Department of Electrical Engineering/ESAT, Katholieke Universiteit Leuven, Heverlee, Belgium

Synonyms
Advanced encryption standard; AES

Related Concepts
►Block Ciphers

Definition
Rijndael is a block cipher that provides a mapping from plaintext blocks to ciphertext blocks and vice versa under a cipher key.

Background
On October 2, 2000, NIST officially announced that Rijndael, without modifications, would become the AES [6]. Rijndael supports all combinations of block lengths and key lengths that are a multiple of 32 bits with a minimum of 128 bits and a maximum of 256 bits. The Rijndael reference specification can be found in ([4], App. E). AES is equal to Rijndael limited to a block length of 128 bits and supports for key length of 128, 192, or 256 bits. AES is specified in [5].

Theory
The most important feature of Rijndael is its consistent good performance among a wide range of platforms.

On smartcards, Rijndael can be implemented using less than 1 kb of code (tables included), and using 36 bytes of memory. Since the text input and the key take both 16 bytes, only 4 bytes extra are required for temporary variables. On the other hand, on high-end processors, Rijndael can exploit cache and parallelism to achieve a significant speedup.

In order to allow fast key setup times, Rijndael has a lightweight key schedule. Fast key setup times are important in systems that switch keys often such as financial authorization schemes or IP security.

Rijndael's structure and choice of operations facilitates implementations that are resistant against side channel attacks.

These features are a result of the application of the following design principles:

Keep it simple: No complexity is added unless there is a demonstrated need for it. The aim is to have a design that is secure against known attacks, without introducing new vulnerabilities. There is no reason to go beyond that, i.e., to add an excessive amount of extra layers of complexity in the hope that this will provide extra security.

Modularity: The design is composed of different building blocks, or steps, each with their own functionality. Building blocks are selected according to specific quantitative selection criteria.

Symmetry and parallelism : All steps can be parallelized and act in a symmetrical way on the input. The large degree of parallelism allows to tradeoff area for speed in a flexible way in hardware implementations.

Choice of operations : All steps are defined with operations in $GF(2^8)$ and can be implemented using XOR and table lookup instructions only. The fact that no arithmetic operations are used, saves space on hardware platforms. The use of operations gives the programmer a lot of flexibility for implementing platform-dependent optimizations on a wide range of processors.

An important factor in the design of Rijndael is the wide trail strategy. This strategy defines diffusion and nonlinearity criteria for the building blocks of the cipher to provide high resistance against differential and linear cryptanalysis in an efficient way. For a detailed treatment, we refer to ([4] Chap. 9).

Structure

When encrypting, the bytes of a plaintext block are mapped onto the elements of a state, the state undergoes a transformation and the elements of the state are mapped onto the bytes of a ciphertext block. Rijndael is a key-iterated block cipher: The transformation from plaintext to ciphertext can be seen as the repeated application of an invertible round transformation, alternated with the addition of round keys. The number of rounds is denoted by N_r. An encryption consists of an initial key addition, denoted by `AddRoundKey`, followed by $\mathrm{N}_r - 1$ applications of the transformation `Round`, and finally one application of `FinalRound`. The initial key addition and every round take as input the `State` and a round key. The round key for round i is denoted by `ExpandedKey`$[i]$, and `ExpandedKey`$[0]$ denotes the input of the initial key addition. The derivation of `ExpandedKey` from the `CipherKey` is denoted by `KeyExpansion`. A high-level algorithm for encryption with Rijndael in pseudo-c notation is shown below.

The number of rounds depends on the block length and the key length. Let N_b be the block length divided by 32 and N_k the length of the cipher key divided by 32. Then:

$$\mathrm{N}_r = \max(\mathrm{N}_k, \mathrm{N}_b) + 6. \quad (1)$$

The encryption and decryption algorithms of Rijndael are not the same, but do have the same structure. For a treatment of these aspects, see [4] Sect. 3.7.

The Round Transformation

The round transformation is denoted `Round`, and is a sequence of four invertible transformations. This is shown in List 2. The final round of the cipher is slightly different: With respect to the round transformation, the `MixColumns` transformation has been removed. It is denoted `FinalRound` and the Rijndael round transformation shown below

The state is a rectangular array of elements of $\mathrm{GF}(2^8)$ of four rows and N_b columns. A byte with bits $b_7b_6b_5b_4b_3b_2b_1b_0$ maps to an element in $\mathrm{GF}(2^8)$ given by the following polynomial:

$$b(x) = b_7x^7+b_6x^6+b_5x^5+b_4x^4+b_3x^3+b_2x^2+b_1x+b_0. \quad (2)$$

where the coefficients b_i are elements of $\mathrm{GF}(2)$ (i.e., bits). In this representation, addition consists of addition of polynomials and multiplication corresponds with multiplication of polynomials modulo the following irreducible polynomial:

$$m(x) = x^8 + x^4 + x^3 + x + 1. \quad (3)$$

In the following, constants in $\mathrm{GF}(2^8)$ are denoted by the hexadecimal notation of the corresponding byte value. For example, `57` corresponds with bit string 01010111 and hence with the polynomial $x^6 + x^4 + x^2 + x + 1$.

`SubBytes` is the only nonlinear transformation of the round. It is a bricklayer permutation consisting of an invertible S-box applied to the elements of the state. Figure 1 illustrates the effect of `SubBytes` on the state.

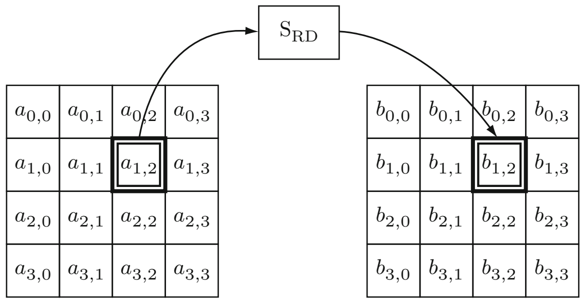
The same S-box is used for all byte positions. This is a design choice motivated by concerns of simplicity and

List. 1 High-level algorithm for encryption with Rijndael

```
Rijndael(State,CipherKey)
{
KeyExpansion(CipherKey,ExpandedKey);
AddRoundKey(State,ExpandedKey[0]);
for(i = 1; i < Nr ; i++) Round(State,
      ExpandedKey[i]);
FinalRound(State,ExpandedKey[Nr]);
```

List. 2 The Rijndael round transformation

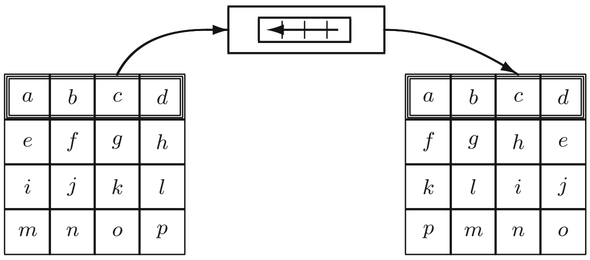
```
Round(State,ExpandedKey[i])
{
SubBytes(State);
ShiftRows(State);
MixColumns(State);
AddRoundKey(State,ExpandedKey[i]);
}

FinalRound(State,ExpandedKey[Nr])
{
SubBytes(State);
ShiftRows(State);
AddRoundKey(State,ExpandedKey[Nr]);
}
```



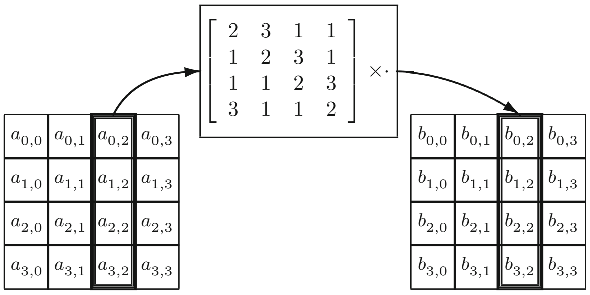
Rijndael. Fig. 1 `SubBytes` acts on the individual bytes of the state

Rijndael. Table 1 `ShiftRows`: shift offsets for different block lengths

block length	C_0	C_1	C_2	C_3
128	0	1	2	3
160	0	1	2	3
192	0	1	2	3
224	0	1	2	4
256	0	1	3	4



Rijndael. Fig. 2 `ShiftRows` operates on the rows of the state



Rijndael. Fig. 3 `MixColumns` operates on the columns of the state

implementation cost. The S-box is defined by the following function in GF(2^8):

$$f_{RD}(x) = \texttt{05} \cdot x^{254} + \texttt{09} \cdot x^{253} + \texttt{F9} \cdot x^{251} + \texttt{25} \cdot x^{247} + \texttt{F4} \cdot x^{239} + \texttt{01} \cdot x^{223} + \texttt{B5} \cdot x^{191} + \texttt{8F} \cdot x^{127} + \texttt{63}\ . \tag{4}$$

The S-box was constructed as the composition of two invertible mappings:

- Multiplicative inverse: To have the desired nonlinearity properties as required by the wide trail strategy. This choice was inspired by K. Nyberg in [7].
- Affine mapping: To complicate the algebraic expression without affecting the nonlinearity properties. This was inspired by algebraic attacks such as interpolation attacks.

In hardware and software, the S-box can be implemented as a lookup table with 256 entries. In hardware the area taken by an S-box can be reduced by exploiting the internal structure of the S-box.

`ShiftRows` is a transposition that cyclically shifts the rows of the state, each over a different offset, as imposed by the wide trail strategy. Row 0 is shifted over C_0 bytes, row 1 over C_1 bytes, row 2 over C_2 bytes, and row 3 over C_3 bytes. The shift offsets C_2 and C_3 depend on the block length. The different values are specified in Table 1. Figure 2 illustrates the effect of `ShiftRows` on the state.

`MixColumns` is a bricklayer permutation operating on the state column by column. The columns of the state are considered as polynomials over GF(2^8) and multiplied modulo $x^4 + 1$ with a fixed polynomial $c(x)$:

$$c(x) = \texttt{03} \cdot x^3 + \texttt{01} \cdot x^2 + \texttt{01} \cdot x + \texttt{02}. \tag{5}$$

This polynomial has been selected as one of the simplest polynomial that has a branch number equal to 5. The branch number is a measure that expresses the diffusion power of a mapping in the context of the wide trail strategy. As illustrated in Fig. 3, the modular multiplication with a fixed polynomial can be written as a matrix multiplication.

The polynomial $c(x)$ is coprime to $x^4 + 1$ and therefore has an inverse modulo $x^4 + 1$. The inverse polynomial $d(x)$ is given by:

$$d(x) = \texttt{0B} \cdot x^3 + \texttt{0D} \cdot x^2 + \texttt{09} \cdot x + \texttt{0E}. \tag{6}$$

In hardware implementations, these linear maps can be efficiently hardwired. In software implementations, table lookups can be used to efficiently exploit a wide range of processors. On 32-bit processors, the sequence of `SubBytes`, `ShiftRows` and `MixColumns` can be implemented by 16 table lookups.

`AddRoundKey` is the key addition. In this transformation, the state is modified by adding a round key to it. The addition in $GF(2^8)$ corresponds with the bitwise XOR operation. The round key length is equal to the block length.

Key Schedule

The key schedule consists of two components: the key expansion and the round key selection. The key expansion specifies how `ExpandedKey` is derived from the cipher key.

The expanded key is can be seen as a rectangular array with four rows of elements in $GF(2^8)$. The key expansion function depends on the key length: There is a version for keys up to 224 bits and a version for keys longer than 224 bits. The detailed specification can be found in [4], Sect. 3.6. In both versions of the key expansion, the first N_k columns of the expanded key are filled with the cipher key. The following columns are computed recursively in terms of previously defined columns. The recursion uses the elements of the previous column, the bytes of the column N_k positions earlier, and round constants $RC[j]$. The round constants are independent of N_k and defined by $RC[j] = x^{j-1}$. The recursive key expansion allows on-the-fly computation of round keys on memory-constrained platforms.

Security

In 2009, a chosen-key distinguisher and a related-key attack were shown for AES-256 [3]. The related-key attack works for a fraction of 2^{-35} of the keys. Quickly thereafter, the results were improved to related-key attacks that work for all keys, on AES-256 and AES-192 [2]. The attack on AES-256 has both data complexity and computational complexity equal to 2^{100}. The attack on AES-192 has a data complexity of 2^{123} and a computational complexity of 2^{176}. There is no consensus on the conclusions that should be drawn from these related-key attacks. More particularly, it has been shown that certain classes of related-key attacks apply even to ideal ciphers [1] and at least some of the related-key attacks on AES appear to belong to such a class.

Applications

Rijndael and AES are used to protect the confidentiality of data in many commercial products. Like with all block ciphers, they are used in well-defined modes of operation, e.g., the Counter (CTR) Mode.

The Rijndael round transformation and parts of it are used as building blocks in many other cryptographic primitives designed after 2000.

Recommended Reading

1. Bellare M, Kohno T (2003) A theoretical treatment of related-key attacks: RKA-PRPs, RKA-PRFs, and applications. In: EUROCRYPT 2003, Warsaw, pp 491–506
2. Biryukov A, Khovratovich D (2009) Related-key cryptanalysis of the FullAES-192 and AES-256. In: Asiacrypt'09, Tokyo. LNCS, vol 5912. Springer, Berlin
3. Biryukov A, Khovratovich D, Nikoliç I (2009) Distinguisher and related- key attack on the full AES-256. In: CRYPTO'09, Santa Barbara. LNCS, vol 5677. Springer, Berlin, pp 231–249
4. Daemen J, Rijmen V (2002) The design of Rijndael. AES – advanced encryption standard. Information security and cryptography. Springer, Heidelberg
5. National Bureau of Standards (Nov 2001) Advanced encryption standard (AES), Federal information processing standard (FIPS), Publication 197. U.S. Department of Commerce, Washington, DC
6. Nechvatal J, Barker E, Bassham L, Burr W, Dworkin M, Foti J, Roback E (2000) Report on the development of the advanced encryption standard (AES), Computer Security Division, Information Technology Laboratory, NIST, Technology Administration, U.S. Department of Commerce
7. Nyberg K (1994) Differentially uniform mappings for cryptography. In: Helleseth T (ed) Advances in cryptology, proceedings Eurocrypt'93, Lofthus. LNCS, vol 765. Springer, Berlin, pp 55–64

Ring

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

Related Concepts

►Field; ►Group

Definition

A *ring* is a set of elements with a certain well defined mathematical structure under two *group operations*.

Theory

A *ring* $R = (S, +, \times)$ is the extension of a ►group $(S, +)$ with an additional operation $\times$, subject to the following additional axioms:

- **Commutativity of** $+$: For all $x, y \in S$, $x + y = y + x$.
- **Closure of** $\times$: For all $x, y \in S$, $x \times y \in S$.
- **Associativity of** $\times$: For all $x, y, z \in S$, $(x \times y) \times z = x \times (y \times z)$.
- **Distributivity of** $\times$ **over** $+$: For all $x, y, z \in S$, $x \times (y + z) = (x \times y) + (x \times z)$ and $(x + y) \times z = (x \times z) + (y \times z)$.

In other words, a ring can be viewed as the extension of a commutative additive group with a multiplication operation. The rings of interest in cryptography generally also have an identity element:

- **Identity of** $\times$: There exists a multiplicative *identity element*, denoted 1, such that for all $x \in S$, $x \times 1 = 1 \times x = x$.

Let S^* denote the elements that have a multiplicative inverse; these are sometimes called the *units* of the ring. The additive identity 0 does not have a multiplicative inverse. Then $(S^*, \times)$ is a group with respect to the multiplication operation; it is called the *multiplicative group* of the ring, or sometimes the *group of units*. Nonzero elements x for which there exist a nonzero element y such that $x \times y = 0$ are called the *zero divisors* of the ring.

A ring is *commutative* if the multiplicative group is also commutative, i.e., for all $x, y \in S$, $x \times y = y \times x$. (The additive group of the ring is always commutative as noted above.)

Applications

The most common ring in public-key cryptography is the ring of integers modulo a composite number n. Here, the ring, denoted Z_n, consists of the set of integers (i.e., residue classes) modulo n and the ring operations are modular addition and multiplication (►Modular Arithmetic). The multiplicative group, denoted Z_n^*, consists of the integers relatively prime to the modulus n, and its ►order is $\phi(n)$, where ϕ is ►Euler's totient function. For instance, if $n = pq$ where p and q are distinct primes, then $\phi(n) = (p-1)(q-1)$.

It is easy to determine $\phi(n)$ given the primes p and q and difficult without them; this fact is one basis for the security of the ►RSA problem.

RIPEMD Family

Antoon Bosselaers
Department of Electrical Engineering, Katholieke Universiteit Leuven, Leuven-Heverlee, Belgium

Related Concepts

►Collision Resistance; ►Hash Functions; ►Preimage Resistance; ►Second Preimage Resistance

The RIPEMD Family designates a family of five different ►Hash Functions: RIPEMD, RIPEMD-128, RIPEMD-160, RIPEMD-256, and RIPEMD-320 [1, 2]. They take variable length input messages and hash them to fixed-length outputs. They all operate on 512-bit message blocks divided into sixteen 32-bit words. RIPEMD (later replaced by RIPEMD-128/160) and RIPEMD-128 produce a hash value of 128 bits, RIPEMD-160, RIPEMD-256, and RIPEMD-320 have a hash result of 160, 256, and 320 bits, respectively. All the five functions start by padding the message according to the so-called Merkle–Damgård strengthening technique (refer to ►Hash Functions for more details). Next, the message is processed block by block by the underlying compression function. This function initializes an appropriate number of 32-bit chaining variables to a fixed value to hash the first message block, and to the intermediate hash value for the following message blocks.

In RIPEMD, RIPEMD-128, and RIPEMD-160, two copies are made from the chaining variables, and both these sets of line variables are processed independently by two parallel lines. Each step of such a parallel line updates in turn one of the line variables using a different message word W_x. After 16 steps, all message words have been used once, and are reused in the next 16 steps, but in a different order. This is repeated three, four, or five times depending on the algorithm. In the last step, the initial values of the chaining variables are combined with both sets of line variables to form the intermediate hash value. When all consecutive message blocks have been hashed, the last intermediate hash value is the hash value for the entire message. RIPEMD-256 and RIPEMD-320 are derived from RIPEMD-128 and RIPEMD-160, respectively, by turning the line variables into chaining variables and by replacing the combination of line variables at the end by a simple feedforward of the initial values of the chaining variables. In addition, the contents of two chaining variables belonging to different lines are exchanged after every 16 steps. The following provides an overview of RIPEMD-160, RIPEMD-128, and their twins RIPEMD-320 and RIPEMD-256.

Padding: The message is appended with a binary one and right-padded with a variable number of zeros followed by the length of the original message (modulo 2^{64}) coded over two binary words. The total padded message length must be a multiple of the message block size.

Initial Values: The RIPEMD Family uses up to ten 32-bit initial values defined as follows:

$$\begin{array}{lcl lcl}
IV_0 & = & 67452301_x & IV_5 & = & 76543210_x \\
IV_1 & = & EFCDAB89_x & IV_6 & = & FEDCBA98_x \\
IV_2 & = & 98BADCFE_x & IV_7 & = & 89ABCDEF_x \\
IV_3 & = & 10325476_x & IV_8 & = & 01234567_x \\
IV_4 & = & C3D2E1F0_x & IV_9 & = & 3C2D1E0F_x
\end{array}$$

RIPEMD-160 Compression Function: Five 32-bit chaining variables h_0, h_1, h_2, h_3, h_4 are either initialized to the fixed values IV_0 through IV_4 for the first 512-bit message block or to the intermediate hash value for the following message blocks. Let "$X^{\lll n}$" represent the cyclic rotation of X to the left by n bits, and let "+" represents addition modulo 2^{32}. Then the compression function works as follows:

```
$A \leftarrow h_0,\ A' \leftarrow h_0$
$B \leftarrow h_1,\ B' \leftarrow h_1$
$C \leftarrow h_2,\ C' \leftarrow h_2$
$D \leftarrow h_3,\ D' \leftarrow h_3$
$E \leftarrow h_4,\ E' \leftarrow h_4$

for $i = 0$ to $79$ do
    $T \leftarrow (A + f_i(B, C, D,) + W_{r(i)} + K_i)^{<<S_i(r(i))} + E$
    $T \leftarrow (A' + f_{79-i}(B', C', D') + W_{r'(i)}$
        $+K_i')^{<<s_i(r'(i))} + E'$

    $A \leftarrow E, A' \leftarrow E'$
    $E \leftarrow D, E' \leftarrow D'$
    $D \leftarrow C^{<<10}, D' \leftarrow C'^{<<10}$
    $C \leftarrow B, C' \leftarrow B'$
    $B \leftarrow T, B' \leftarrow T'$
$T \leftarrow h_1 + C + D'$
$h_1 \leftarrow h_2 + D + E'$
$h_2 \leftarrow h_3 + E + A'$
$h_3 \leftarrow h_4 + A + B'$
$h_4 \leftarrow h_0 + B + C'$
$h_0 \leftarrow T$
```

where the ordering of message words $r(i)$ and $r'(i)$, the nonlinear functions f_i, the shifts s_i, and the constants K_i and K_i' are defined as:

1. **Ordering of the message words $r(i)$ and $r'(i)$.** Take the following permutation ρ:

j	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$\rho(j)$	7	4	13	1	10	6	15	3	12	0	9	5	2	14	11	8

Further define the permutation π by setting $\pi(j) = (9j + 5) \bmod 16$. The ordering of the message words $r(i)$ and $r'(i)$ is then given by the following table:

	$0 \le i \le 15$	$16 \le i \le 31$	$32 \le i \le 47$	$48 \le i \le 63$	$64 \le i \le 79$
$r(i)$	i	$\rho(i-16)$	$\rho^2(i-32)$	$\rho^3(i-48)$	$\rho^4(i-64)$
$r'(i)$	$\pi(i)$	$\rho\pi(i-16)$	$\rho^2\pi(i-32)$	$\rho^3\pi(i-48)$	$\rho^4\pi(i-64)$

2. **Nonlinear functions f_i.** Let "$\oplus$," "$\vee$," "$\wedge$," and "$\neg$" represent, respectively, bit-wise exclusive-or, bit-wise or, bit-wise and, and bit-wise complement:

$$
\begin{aligned}
f_i(x, y, z) &= x \oplus y \oplus z, & 0 \le i \le 15\\
f_i(x, y, z) &= (x \wedge y) \vee (\neg x \wedge z), & 16 \le i \le 31\\
f_i(x, y, z) &= (x \vee \neg y) \oplus z, & 32 \le i \le 47\\
f_i(x, y, z) &= (x \wedge z) \vee (y \wedge \neg z), & 48 \le i \le 63\\
f_i(x, y, z) &= x \oplus (y \vee \neg z), & 64 \le i \le 79
\end{aligned}
$$

3. **Shifts $s_i(j)$**

j	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$0 \le i \le 15$	11	14	15	12	5	8	7	9	11	13	14	15	6	7	9	8
$16 \le i \le 31$	12	13	11	15	6	9	9	7	12	15	11	13	7	8	7	7
$32 \le i \le 47$	13	15	14	11	7	7	6	8	13	14	13	12	5	5	6	9
$48 \le i \le 63$	14	11	12	14	8	6	5	5	15	12	15	14	9	9	8	6
$64 \le i \le 79$	15	12	13	13	9	5	8	6	14	11	12	11	8	6	5	5

4. **Constants K_i and K_i'.**

	$0 \le i \le 15$	$16 \le i \le 31$	$32 \le i \le 47$	$48 \le i \le 63$	$64 \le i \le 79$
K_i	00000000_x	$5A827999_x$	$6ED9EBA1_x$	$8F1BBCDC_x$	$A953FD4E_x$
K_i'	$50A28BE6_x$	$5C4DD124_x$	$6D703EF3_x$	$7A6D76E9_x$	00000000_x

RIPEMD-128 Compression Function: The main difference with RIPEMD 160 is that a hash result and chaining variable of 128 bits (four 32-bit words) is used and that there are only 64 steps. Four 32-bit chaining variables h_0, h_1, h_2, h_3 are either initialized to the fixed values IV_0 through IV_3 for the first 512-bit message block or to the intermediate hash value for the following message blocks. Then the compression function works as follows:

```
$A \leftarrow h_0,\ A' \leftarrow h_0$
$B \leftarrow h_1,\ B' \leftarrow h_1$
$C \leftarrow h_2,\ C' \leftarrow h_2$
$D \leftarrow h_3,\ D' \leftarrow h_3$
for $i = 0$ to $63$ do
    $T \leftarrow (A + f_i(B, C, D) + W_{r(i)} + K_i)^{<<s_i(r(i))}$
    $T' \leftarrow (A' + f_{63-i}(B', C', D') + W_{r'(i)}$
        $+K_i')^{<<s_i(r'(i))}$
    $A \leftarrow D, A' \leftarrow D'$
    $D \leftarrow C, D' \leftarrow C'$
    $C \leftarrow B, C' \leftarrow B'$
    $B \leftarrow T, B' \leftarrow T'$
$T \leftarrow h_1 + C + D'$
$h_1 \leftarrow h_2 + D + A'$
$h_2 \leftarrow h_3 + A + B'$
$h_3 \leftarrow h_0 + B + C'$
$h_0 \leftarrow T$
```

where the ordering of message words $r(i)$ and $r'(i)$, the nonlinear functions f_i, the shifts s_i, and the constants K_i and K_i' are defined as in RIPEMD-160, except that $K_i' = 00000000_x$ for $48 \le i \le 63$.

RIPEMD-320 Compression Function: Ten 32-bit chaining variables $h_0, h_1, h_2, h_3, h_4, h_5, h_6, h_7, h_8, h_9$ are either initialized to the fixed values IV_0 through IV_9 for the first 512-bit message block or to the intermediate hash value for the following message blocks. Then the compression function works as follows:

```
$A \leftarrow h_0,\ A' \leftarrow h_5$
$B \leftarrow h_1,\ B' \leftarrow h_6$
$C \leftarrow h_2,\ C' \leftarrow h_7$
$D \leftarrow h_3,\ D' \leftarrow h_8$
$E \leftarrow h_4,\ E' \leftarrow h_9$
for $i = 0$ to $79$ do
  $T \leftarrow (A + f_i(B, C, D) + W_{r(i)} + K_i)^{<<s_i(r(i))}$
      $E$
  $T \leftarrow (A' + f_{79-i}(B', C', D') + W_{r'(i)}$
      $+K_i')^{<<s_i(r'(i))} + E'$
  $A \leftarrow E, A' \leftarrow E'$
  $E \leftarrow D, E' \leftarrow D'$
  $D \leftarrow C^{<<10}, D' \leftarrow C'^{<<10}$
  $C \leftarrow B, C' \leftarrow B'$
  $B \leftarrow T, B' \leftarrow T'$
  if $i = 15$ then
  $T \leftarrow B, B \leftarrow B', B' \leftarrow T$
  else if $i = 31$ then
  $T \leftarrow D, D \leftarrow D', D' \leftarrow T$
  else if $i = 47$ then
  $T \leftarrow A, A \leftarrow A', A' \leftarrow T$
  else if $i = 63$ then
  $T \leftarrow C, C \leftarrow C', C' \leftarrow T$
  else if $i = 79$ then
  $T \leftarrow E, E \leftarrow E', E' \leftarrow T$
$h_0 \leftarrow h_0 + A, h_5 \leftarrow h_5 + A'$
$h_1 \leftarrow h_1 + B, h_6 \leftarrow h_6 + B'$
$h_2 \leftarrow h_2 + C, h_7 \leftarrow h_7 + C'$
$h_3 \leftarrow h_3 + D, h_8 \leftarrow h_8 + D'$
$h_4 \leftarrow h_4 + E, h_9 \leftarrow h_9 + E'$
```

where the ordering of message words $r(i)$ and $r'(i)$, the nonlinear functions f_i, the shifts s_i, and the constants K_i and K_I' are defined as in RIPEMD-160.

RIPEMD-256 Compression Function: The main difference with RIPEMD 320 is that a hash result and chaining variable of 256 bits (eight 32-bit words) is used and that there are only 64 steps. Eight 32-bit chaining variables $h_0, h_1, h_2, h_3, h_4, h_5, h_6, h_7$ are either initialized to the fixed values IV_0 through IV_3 and IV_5 through IV_8 for the first 512-bit message block or to the intermediate hash value for the following message blocks. Then the compression function works as follows:

```
$A \leftarrow h_0, A' \leftarrow h_4$
$B \leftarrow h_1, B' \leftarrow h_5$
$C \leftarrow h_2, C' \leftarrow h_6$
$D \leftarrow h_3, D' \leftarrow h_7$
for $i = 0$ to $63$ do
  $T \leftarrow (A + f_i(B, C, D) + W_{r(i)} + K_i)^{\ll s_i(r(i))}$
  $T' \leftarrow (A' + f_{63-i}(B', C', D') + W_{r'(i)}$
      $+K_i')^{\ll s_i(r'(i))}$
  $A \leftarrow D, A' \leftarrow D'$
  $D \leftarrow C, D' \leftarrow C'$
  $C \leftarrow B, C' \leftarrow B'$
  $B \leftarrow T, B' \leftarrow T'$
  if $i = 15$ then
  $T \leftarrow A, A \leftarrow A', A' \leftarrow T$
  else if $i = 31$ then
  $T \leftarrow B, B \leftarrow B', B' \leftarrow T$
  else if $i = 47$ then
  $T \leftarrow C, C \leftarrow C', C' \leftarrow T$
  else if $i = 63$ then
  $T \leftarrow D,\ D \leftarrow D', D' \leftarrow T$
  $h_0 \leftarrow h_0 + A, h_4 \leftarrow h_4 + A'$
  $h_1 \leftarrow h_1 + B, h_5 \leftarrow h_5 + B'$
  $h_2 \leftarrow h_2 + C, h_6 \leftarrow h_6 + C'$
  $h_3 \leftarrow h_3 + D, h_7 \leftarrow h_7 + D'[3pt]$
```

where the ordering of message words $r(i)$ and $r'(i)$, the nonlinear functions f_i, the shifts s_i, and the constants K_i and K_i' are defined as in RIPEMD-160, except that $K_i' = 00000000_x$ for $48 \le i \le 63$.

RIPEMD: The original RIPEMD consists of essentially two parallel versions of MD4, with some improvements to the shifts and the order of the message words; the two parallel instances differ only in the round constants. At the end of the compression function, the words of left and right halves are added to each other and to the initial values of the chaining variable.

Security Considerations: The RIPEMD Family has been designed to provide ►Collision Resistance. RIPEMD was developed in 1992 in the framework of the EC-RACE project RIPE [1]. In 1995, Dobbertin found collisions for reduced versions of RIPEMD [3]. Due to these partial attacks, RIPEMD was upgraded in 1996 by Dobbertin et al. to RIPEMD-128 (as plug-in substitute for RIPEMD) and RIPEMD-160 [2]. At the same time, the variants RIPEMD-256 and RIPEMD-320 were introduced as well. An additional reason for the introduction of RIPEMD-160 is brute force collision search attacks. In [4], van Oorschot and Wiener estimate that with a ten million US$ machine collisions of MD5 can be found in 21 days in 1994, which corresponds to 4 h in 2004. To counter such collision search attacks, hash values of at least 160 bits are required. RIPEMD-128 and RIPEMD-160 are included in ISO/IEC 10118–3 [5].

Recommended Reading

1. RIPE (1995) Integrity primitives for secure information systems. In: Bosselaers A, Preneel B (eds) Final Report of RACE Integrity Primitives Evaluation (RIPE-RACE 1040). Lecture notes in computer science, vol 1007. Springer, Berlin
2. Dobbertin H, Bosselaers A, Preneel B (1996) RIPEMD-160: a strengthened version of RIPEMD. In: Gollmann D (ed) Fast Software Encryption, Cambridge, UK, 21–23 February 1996. Lecture notes in computer science, vol 1039. Springer, Berlin, pp 71–82. Final version available at http://www.esat.kuleuven.ac.be/~cosicart/pdf/AB-9601/. More information on all aspects of RIPEMD-xxx can be found at http://www.esat.kuleuven.ac.be/~bosselae/ripemd160/
3. Dobbertin H (1992) RIPEMD with two-round compress function is not collisionfree. J Cryptol 10(1):51–69
4. van Oorschot PC, Wiener M (1999) Parallel collision search with cryptanalytic applications. J Cryptol 12(1):1–28
5. ISO/IEC 10118-3 (2003) Information technology—security techniques—hash-functions—Part 3: Dedicated hash-functions

Rivest Cipher 5

►RC5

Rivest Cipher 6

►RC6

Role-Based Access Control

Vijay Alturi[1], David Ferraiolo[2]
[1]Management Science and Information Systems Department, Center for Information Management, Integration and Connectivity, Rutgers University, Newark, NJ, USA
[2]National Institute of Standards and Technology, Gaithersburg, MD, USA

Synonyms

RBAC

Related Concepts

►Access Control; ►Access Control, Policies, Models, and Mechanisms; ►Roles in SQL

Definition

Roles represent organizational agents that perform certain job functions within the organization. RBAC controls all access to system resources through roles assigned to users. A role is assigned (directly or indirectly through inheritance) a collection of permissions that are made available to users who have been assigned to the role.

Background

The concept of roles has been used in software applications for more than 3 decades, which originated from groups in operating systems and privilege grouping in database management systems. By combining the Ferraiolo–Kuhn model (1992) and the framework proposed by 1996 Sandhu et al. (1996), in 2004, the National Institute of Standards and Technology (NIST) proposed a US national standard for role-based access control (RBAC) through the International Committee for Information Technology Standards (ANSI/INCITS). The most striking aspect of RBAC's history is its rapid evolution from a concept to its commercial implementation and deployment. This success can be attributed to the fact that the cost of deployment need not be justified solely on perceived threats and vulnerabilities alone, but also based on economic benefits. RBAC helps to reduce the complexity of security administration and supports the review of permissions assigned to users. This latter aspect is sometimes referred to as "before the fact auditing." Today, RBAC has been adopted successfully by a variety of commercial systems. As a result, RBAC has become the norm in many of today's organizations for enforcing security. In addition to its commercial success, the research community has been highly active, published hundreds of papers, and even started an ACM workshop in 1995, which evolved now into an ACM Symposium. The primary aspects of the research are formalizing the RBAC model and its design, and extending the model to various environments including temporal, spatial, mobile databases, workflow systems, web services, social networks, collaborative systems, etc.

Theory

The NIST RBAC standard defines 4 partially ordered categorizations – Core RBAC, Hierarchical RBAC, and Constrained RBAC. These are depicted in Fig. 1, and can be formally (static and dynamic) defined as follows.

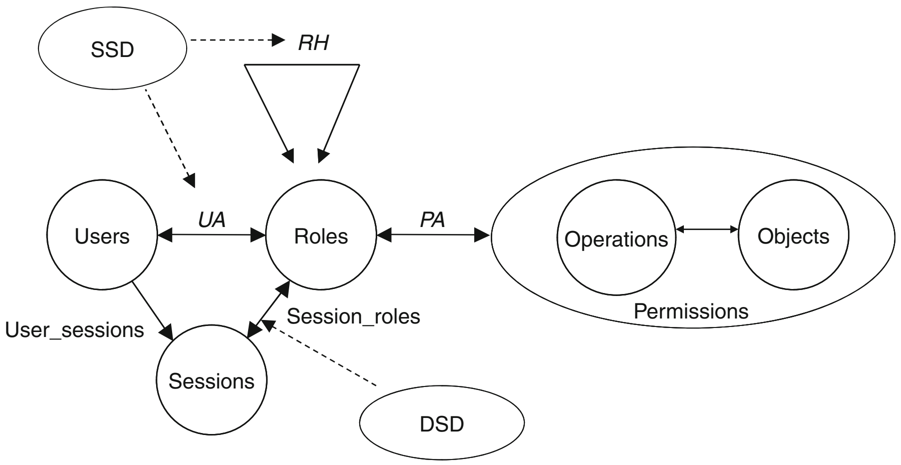
Definition 1

- Core RBAC
  - U, R, OP, and O are the set of users, roles, operations, and objects
  - $UA \subseteq U \times R$, a many-to-many mapping user-to-role assignment relation
  - $P = 2^{(OP \times O)}$, the set of permissions
  - $PA \subseteq P \times R$, a many-to-many mapping of permission-to-role assignments
  - $UPA \subseteq U \times P$, a many-to-many mapping of user-to-permission assignments
  - $assigned\_users(R) = \{u \in U | (u, R) \in UA\}$, the mapping of role R onto a set of users
  - $assigned\_permissions(R) = \{p \in P | (p, R) \in PA\}$, the mapping of role R onto a set of permissions
  - S, the set of sessions
  - $user\_sessions(u : U) \to 2^S$, the mapping of user u onto S
  - $session\_roles(s : S) \to 2^R$, the mapping of session s onto R
  - $avail\_session\_perms(s : S) \to 2^P$, the permissions available to a user in a session, $\bigcup_{r \in session\_roles(s)}$ assig $ned\_permissions(r)$
- Hierarchical RBAC
  - $RH \subseteq R \times R$ is a partial order on R called the inheritance relation, where $r_2 \geq r_1 \Rightarrow authorized\_permissions(r_2) \subseteq authorized\_permissi$ $ons\ (r_1) \wedge authorized\_users(r_2) \subseteq authorized\_users$ (r_1).
- Constrained RBAC
  - Static Separation of Duty (SSD): $SSD \subseteq (2^R \times N)$ is a collection of pairs (rs, n) in SSD, where each rs is a role set, and n is a natural number ≥ 2 with the property that no user is assigned to n or more roles from the set rs in each $(rs, n) \in SSD$.
  - Dynamic Separation of Duty (DSD): $DSD \subseteq (2^R \times N)$ is a collection of pairs (rs, n) in DSD, where each rs is a role set, and n is a natural number ≥ 2 with the property that no user is assigned to n or more roles from the set rs in a single session in each $dsd \in DSD$.

In addition, NIST RBAC standard defines administrative functions pertaining to creation and maintenance of the basic elements and relations of RBAC, and permission review functions.

Applications

Today, RBAC is being widely deployed at almost all levels of enterprise computing and infrastructure, including operating systems, database management systems,



Role-Based Access Control. Fig. 1 RBAC elements and relations network operating systems, workflow management systems, web services, and collaborative and virtual enterprise systems.

Experimental Results

A study by the NIST conducted in 2002 by Research Triangle Institute (RTI) through interviews with software developers and end users of RBAC products concluded that the activities of the NIST RBAC project "accelerated the development and adoption of RBAC by 1 year and "lowered R&D costs for software vendors by approximately six percent."

Using the key metrics of administrative savings associated with managing access to information systems and reduced employee downtime from waiting to receive system access, this study found that NIST's infrastructure technologies and associated infrastructure support efforts result in significant increase in benefit–cost ratio.

Open Problems and Future Directions

Although RBAC has been successfully adopted by many organizations, devising a complete and meaningful set of roles still remains a major challenge. Automated and efficient techniques are needed to address this problem, also known as role engineering. Additionally, the aspect of role evolution and its implications have not been investigated well. Furthermore, current RBAC model should accommodate for exceptions, which may result in succinct specification of roles.

From the perspective of expressing access control policies, there may be certain permissions that may not be part of a role and therefore cannot be squeezed under the RBAC framework. The question then is how identity-based access control can coexist with RBAC. Under the same vein, there is a need for a rich model capable of expressing a variety of access control policies needed by organizations that need not necessarily be expressed under RBAC. While RBAC has the benefit of facilitating analysis of the permissions possessed by users, alternative models, such as attribute-based access control, do not enjoy that luxury. Models that are capable of rich expressive power and at the same time lend themselves for easy analysis are needed. Issues of RBAC administration and safety analysis have not been adequately studied, especially in the context of the many extensions of RBAC.

Recommended Reading

1. Ferraiolo DF, Kuhn DR, Chandramouli R (2007) Role-based access control. Artech House, Boston, p 338, ISBN 1-59693-113-8, 2007
2. http://csrc.nist.gov/groups/SNS/rbac/

Roles in SQL

Alessandro Campi
Dipartimento di Elettronica e Informazione, Politecnico di Milano, Milano, Italy

Synonyms

RBAC

Related Concepts

►ε-Privacy; ►Role-Based Access Control

Definition

Databases have a number of users viewing and accessing data, which makes security a major concern. SQL roles, which grant and deny permissions to groups of users, are used to control the access to data allowing users to view or modify only the data they are authorized. More precisely, a role defines what a user can and cannot do within a database, and multiple users can share the same role.

Background

Roles are a part of the tiered security model: they can be used to manage login security relating to the server connections, database security getting appropriate accesses to the database and getting appropriate accesses to individual database objects and data. When the user logs in to the server entering a password, access to the stored databases is determined by user accounts. After gaining access to an actual database, the user is restricted to the data he or she can view and modify. The main benefit of roles is efficient management: you can group users needing to view or modify new data and simply select an existing group and assign it to a role instead of modifying all the user accounts.

The Role-Based Access Control (RBAC) model [22] consists of four basic components: users, roles, permissions, sessions. The user can be a real user or a program interacting with the database, a role is a collection of permissions needed to perform a certain job function within an organization, a permission is an access mode that can be exercised on objects in the system, and a session relates a user to possibly many roles. During a session, an user requests activation of a role. An activation request is granted only if the corresponding role is enabled at the time of the request, and the user requesting the activation is entitled to activate the role at the time of the request. If an activation request is satisfied, the user issuing the request obtains all the permissions associated with the role he has requested to activate. In this model, users and permissions are associated with roles to roles. Many users can be authorized to play the same role, and a role can have many permissions, and the same permissions can be associated with many roles. For more details see [17].

Theory or Application, or Both

RBAC is used in OSs and DBMSs. It is an important component in managing data. An important component of any strong security solution is represented by Intrusion Detection (ID) techniques, able to detect anomalous behavior of applications and users. Kamra et al. [15] propose an ID mechanism tailored for DBMS. The approach is based on mining SQL queries stored in database audit log files. The result of the mining process is used to form profiles that can model normal database access behavior and identify intruders. Under a RBAC system, the proposed techniques is able to determine role intruders, that is, individuals while holding a specific role behave differently than expected. An important advantage of providing an ID technique specifically tailored to RBAC databases is that it can help in protecting against insider threats. Osborn et al. [19] describe a methodology for using RBAC design to manage and effect changes to underlying relational database not originally designed to support RBAC. In [5], a formal framework for reasoning about access control models is presented.

Some databases have predefined roles. For example, in SQL Server [9, 13], there are server roles and database roles. Server roles are maintained by the database administrator and apply to the entire server, not an individual database file. The public role sets the basic default permissions for all users. Every user that's added is automatically assigned to the public role. Database roles are applied to an individual database. Roles are managed using the SQL Server Enterprise Manager.

Open Problems

Access control is now applied in new scenarios with a lot of ad-hoc solutions. An interesting extension of RBAC is Temporal-RBAC (TRBAC). TRBAC supports periodic role enabling and disabling expressed by means of role triggers. Role trigger actions may be either immediately executed, or deferred by an explicitly specified amount of time. A formal semantics for TRBAC is described in [4]. It also defines specification language and proposes a system implementing TRBAC.

GRBAC (generalized RBAC) [10] introduces the concept of environment roles, that is, roles that can be activated based on the value of conditions in the environment where the request has been made. X-GTRBAC [7] augments GTRBAC with XML for supporting the policy enforcement in a heterogeneous, distributed environment. A notable approach is generalized TRBAC (GRTBAC) [16] which incorporates language constructs for the specification of various temporal constraints on roles, including constraints on role enabling, role activation, user-to-role assignments, and permission-to-role assignments.

In [24], RBAC model is adapted to object-oriented databases proposing the incorporation of privileges into roles such that roles consist of state, operations, as well as a set of privileges. Frameworks to apply RBAC to XML documents are proposed in [14] and [23]. The proposed methodology is inspired by the work described in [21] designed for object-oriented databases. The work demonstrates how different access modes, XPath expressions, and roles can be combined. In [12], ancestor and sibling relationships are considered as sensitive as the one carried out by the nodes themselves and the integration of relationships as first class citizen in the access control models for XML is advocated.

RBAC model is extended in [1] by considering additional concepts useful to improve authorization control, and by integrating different types of possible links between users of different roles to avoid security breaches in systems devoted to exchanging multimedia objects between wide ranges of distributed applications, web services, and end-users.

The integration of the spatial dimension into RBAC-based models has emerged in advanced GIS and mobile applications. A foundation work describing access-control model for geographical data has been proposed in [2]. The access-control system proposed in [6] introduces the concept of spatial authorization, which can be defined only on portions of space. The system checks if requested objects are in the authorization space before granting access. The work in [20] reaches similar results exploiting XML as spatial data format. A complex spatial data model is uesd in [3]: the access-control system allows the specification of authorization rules to control access to complex-structured spatial data stored in a DBMS and organized according to multiple spatial representation levels and at multiple level of granularity. In [18] is described a proposal that integrates geospatial and security standards to support controlled access to spatial information through geo-Web services. GEO-RBAC [11] uses a geometric model to represent objects, to model user positions, and to assign spatial extents to roles. Roles are activated based on the position of the user. Besides a physical position, obtained from a given mobile terminal or a cellular phone, users are also assigned a logical and device-independent position, representing the feature (the road, the town, the region) in which they are located.

Privacy preservation systems rely on RBAC models. For example, [8] presents a comprehensive approach for privacy preserving access control based on the notion of purpose. Purpose information associated with a given data element specifies the intended use of the data element. This model allows multiple purposes to be associated with each data element and also supports explicit prohibitions, thus allowing privacy officers to specify that some data should not be used for certain purposes.

Recommended Reading

1. Al Bouna B, Chbeir R (2006) Multimedia-based authorization and access control policy specification. In: SWS '06: proceedings of the 3rd ACM workshop on secure web services. ACM, New York, pp 61–68
2. Atluri V, Mazzoleni P (2002) A uniform indexing scheme for geo-spatial data and authorizations. In: Proceedings of the sixteenth conference on data and application security. Cambridge, pp 207–218, http://portal.acm.org/citation.cfm?id=1359363&CFID=13016546&CFTOKEN=95674504
3. Belussi A, Bertino E, Catania B, Damiani ML, Nucita A (2004) An authorization model for geographical maps. In: GIS '04: proceedings of the 12th annual ACM international workshop on geographic information systems. ACM, New York, pp 82–91
4. Bertino E, Bonatti PA, Ferrari E (2001) Trbac: a temporal role-based access control model. ACM Trans Inf Syst Secur 4(3):191–233
5. Bertino E, Catania B, Ferrari E, Perlasca P (2001) A logical framework for reasoning about access control models. In: SACMAT '01: proceedings of the sixth ACM symposium on access control models and technologies. ACM, New York, pp 41–52
6. Bertino E, Damiani ML, Momini D (2004) An access control system for a web map management service. In: RIDE '04: proceedings of the 14th international workshop on research issues on data engineering: web services for e-commerce and e-government applications (RIDE'04). IEEE Computer Society, Washington, DC, pp 33–39
7. Bhatti R, Ghafoor A, Bertino E, Joshi JBD (2005) X-gtrbac: an xml-based policy specification framework and architecture for enterprisewide access control. ACM Trans Inf Syst Secur 8(2):187–227
8. Byun JW, Li N (2008) Purpose based access control for privacy protection in relational database systems. VLDB J 17(4):603–619
9. Chigrik A (2001) Understanding sql server roles. http://www.databasejournal.com/features/mssql/article.php/1441261/understanding-SQL-server-roles.htm. Accessed 15 July 2010
10. Covington MJ, Long W, Srinivasan S, Dev AK, Ahamad M, Abowd GD (2001) Securing context-aware applications using environment roles. In: SACMAT '01: proceedings of the sixth ACM symposium on access control models and technologies. ACM, New York, pp 10–20
11. Damiani ML, Bertino E, Catania B, Perlasca P (2007) Geo-rbac: a spatially aware rbac. ACM Trans Inf Syst Secur 10(1):2
12. Finance B, Medjdoub S, Pucheral P (2005) The case for access control on xml relationships. In: CIKM '05: proceedings of the 14th ACM international conference on Information and knowledge management. ACM, New York, pp 107–114
13. Harkins S (2004) Understanding roles in sql server security. http://articles.techrepublic.com.com/5100-10878_11-1061781.html. Accessed 15 July 2010
14. Hitchens M, Varadharajan V (2001) Rbac for xml document stores. In: ICICS '01: proceedings of the third international conference on information and communications security. Springer, London, pp 131–143
15. Kamra A, Terzi E, Bertino E (2008) Detecting anomalous access patterns in relational databases. VLDB J 17(5):1063–1077
16. Joshi JBD, Bertino E, Latif U, Ghafoor A (2005) A generalized temporal role-based access control model. IEEE Trans Knowl Data Eng 17(1):4–23
17. Ling L, zsu MT (ed) (2009) Encyclopedia of database systems, vol 4. Springer
18. Matheus A (2005) Declaration and enforcement of fine-grained access restrictions for a service-based geospatial data infrastructure. In: SACMAT '05: proceedings of the tenth ACM symposium on Access control models and technologies. ACM, New York, pp 21–28
19. Osborn SL, Han Y, Liu J (2003) A methodology for managing roles in legacy systems. In: SACMAT '03: proceedings of the eighth ACM symposium on access control models and technologies. ACM, New York, pp 33–40
20. Purevjii BO, Amagasa T, Imai S, Kanamori Y (2004) An access control model for geographic data in an xml-based framework. In: WOSIS, pp 251–260
21. Rabitti F, Woelk D, Kim W (1988) A model of authorization for object-oriented and semantic databases. In: EDBT '88: proceedings of the international conference on extending database technology. Springer, London, pp 231–250
22. Sandhu RS (1996) Role hierarchies and constraints for lattice-based access controls. In: ESORICS '96: proceedings of the 4th European symposium on research in computer security. Springer, London, pp 65–79
23. Wang J, Osborn SL (2004) A role-based approach to access control for xml databases. In: SACMAT '04: proceedings of the ninth ACM symposium on access control models and technologies. ACM, New York, pp 70–77
24. Wong RK (1997) Rbac support in object-oriented role databases. In: RBAC '97: proceedings of the second ACM workshop on role-based access control. ACM, New York, pp 109–120

Ron's Code 5

►RC5

Ron's Code 6

►RC6

Root of Trust

William D. Casper, Stephen M. Papa
High Assurance Computing and Networking Labs (HACNet), Department of Computer Science and Engineering, Bobby B. Lyle School of Engineering, Southern Methodist University, Houston, TX, USA

Synonyms

IC-Integrated circuit; TCG trusted computing group; TPM trusted platform module

Related Concepts

►Levels of Trust; ►Trusted Boot; ►Trusted Computing

Definition

Root of Trust: A system element that provides services, including verification of system, software & data integrity and confidentiality, and data (software and information) integrity attestation between other trusted devices in a system or network.

Background

Trusted Computing starts with a Root of Trust, and depending on the required level of trust, the Root of Trust may be software or hardware. In most modern systems, a hardware approach is preferred because it is harder to attack. To establish a high level of trust, there is a need to design the system hardware to ensure the integrity of the hardware, software, and data has not been compromised. The hardware should support monitoring to ensure hardware and software integrity is maintained throughout the power-up, initialization, and run-time operation. To do this, it is extremely important to have at least one hardware device that can act as a Root of Trust within the system. The hardware Root of Trust and any associated firmware or software within the Root of Trust is responsible for establishing and maintaining trust. A commercial example of a Root of Trust is the TPM defined by the TCG [1]. The TPM has architectural requirements for attestation, cryptography, system protection, and ensuring confidentiality and integrity of software and data within the system.

Hardware Root of Trust

A hardware Root of Trust should be capable of verifying its own integrity, and then extend trust by authenticating the expected operating environment (Boot software, Operating System, and other software and hardware elements) within the system. Once it has successfully verified its immediate operating environment, it can then extend trust to other systems. Extension of trust is accomplished using cryptographic authentication or attestation with other Roots of Trust within the system or across a network to other systems. After the extended system is verified, the authenticated software and data can be exchanged and/or loaded and used within the now trusted operating environment.

For networked computers, where the attacker does not have physical access to the hardware, the TPM provides all required Root of Trust services for the system. The TPM ensures the integrity and confidentiality of the software and data while it is sent across networks and when it is stored on the computer. The TPM must verify the integrity of all software and data prior to its use by the system's processing unit or CPU.

When the attacker may have access to the system, there is a need to protect the integrity and confidentiality of data while it is being processed by the CPU. In this case, a secure processor is needed to provide the required root of trust. Trusted devices such as FPGAs with embedded CPU cores are an example of a secure processor that may be used to provide the hardware Root of Trust functionality. In many small or self-contained systems, the secure processor alone may be sufficient to meet all of the processing and protection needs for the sensitive software and data, but in most modern systems, these devices lack the processing throughput to perform all required processing.

Attributes of a TPM Root of Trust

Based on the requirements from the TCG, the key Architectural Elements [1] that a TPM should include to provide a high level of protection against attacks, including the following:

1. Trusted Boot
2. Cryptography including symmetric and asymmetric, hash, digital signature
3. Authentication protocol support
4. Secure Communication/key exchange protocol support
5. Protected Key Storage
6. Attack Protection features
7. Integrity reporting

The PC platform TPM is primarily specified to prevent external/network attacks from being successful, and the mobile platform specifications include additional physical protection requirements and considerations for digital rights management and data protection.

The main difference between a TPM and a secure processor is that the secure processor will also protect the confidentiality of the software and data during run-time execution. A secure processor is a System on a Chip or SoC that is designed to verify and protect the confidentiality and integrity of the software or data prior to and during execution. To do this multiple architecture, attributes are required, including the use of cryptography internal to the device. Some of the challenges include how much cryptographic key material must be securely stored and protected within the Integrated Circuit where the average hacker or even a very sophisticated attacker has trouble accessing the information (tools and knowledge are two major barriers).

Attributes of a Secure Processor Root of Trust

A secure processor should include all of the attributes of the TPM plus others to ensure the confidentiality of the software and data execution. There is a set of key Architectural Elements, based on existing research [2–7], that a secure processor should include to create a root of trust, and to protect integrity and confidentiality of data and software. These elements include the following:

1. One or more CPUs
2. Internal unencrypted memory
3. Hardware to allow low-latency external memory encryption and decryption (confidentiality and integrity)
4. Hardware support for Trusted Boot through OS verification and loading
5. Hardware providing symmetric and asymmetric cryptography, hash, digital signature, etc. (Suite-B algorithms)
6. Support for Authentication protocol support
7. Support for establishing and maintaining Secure Communication channels
8. Support for key exchange protocol
9. Support for Process separation (SW/Data, data register and cache separation)
10. Key storage and Random Number Generation
11. Attack Protection features

There is currently no single commercial device that provides this full complement of functions in a transparent manner, but there are several devices that provide nearly all this functionality. The combination of the secure processor with developed trusted software and operating system will allow for the establishment of trust internal and external to the system and data protection while the data is in storage, in use, and when it is sent across networks.

Existing secure processing devices all have a major drawback. The software and data being protected often exceeds the devices internal memory constraints. This leads to the need for a large main memory external to the device, the need to cryptographically protect the information in the external memory, and the need to store or deterministically generate keys to decrypt the data – in a real-time manner with minimal latency.

Current Research in Secure Processing

Secure processing has been the subject of several academic research projects over the last decade. Two noteworthy projects are XOM (pronounced "ZOM") and AEGIS because these focused on protection of data inside and external to the device, and did not depend on software (including the OS) for protection. XOM was originally proposed in 2000, and AEGIS in 2003.

Open Problems

Secure Processing:
There are multiple research projects identifying architectural attributes of secure processors (XOM and AEGIS), and the Trusted Computing Group is establishing industry-level standards. Unfortunately, the TFG and most commercial processor companies are not focused on a secure monolithic System on Chip (SoC) architecture to support secure processing to support the highest levels of trust. A lot of the existing research on secure processing is out of date or not very relevant to modern multi-core processors or systems. Some potential topics for future research include the following:

1. Extension of trust in systems using TPMs or other Roots of Trust
2. Methods to verify the integrity of ICs within a system to extend trust to the rest of the systems hardware
3. Physically Unclonable Functions
   (a) Reasonable applications methods to generate keys reliably
   (b) Workable challenge response mechanisms
   (c) Support for attestation
4. Control of the JTAG interface by hardware to extend hardware trust, and prevention of the interface for use in a system attack

Run-time Memory Encryption:
In-line memory encryption adds to the memory access bottle neck. Some papers have identified crypto attacks and design guidelines. Most published research select a block encryption scheme like AES and then quantifies the latency for using the scheme or identify methods to minimize this latency. Other research suggests the use of One-Time Pads (OTP) or stream ciphers without details of how to efficiently implement or design the crypto engine into the processor.

There is a real need for additional research on new and efficient real-time methods of encryption and decryption before external memory crypto in real time–embedded or desktop systems will be possible.

Recommended Reading

1. TCG Specifications (2007) Architecture Overview Revision 1.4, 2 Aug 2007. http://www.trustedcomputinggroup.org/files/resource_files/AC652DE1-1D09-3519-ADA026A0C05CFAC2/TCG_1_4_Architecture_Overview.pdf. Mobile Phone Work Group Selected Use Case Analyses – v 1.0. http://www.trustedcomputing

group.org/files/temp/6443B207-1D093519-AD3180491A6DF1F5/MPWG%20Selected_Mobile_Phone_Use_Case_Analyses_v1.pdf
2. Rogers B, Solihin Y, Prvulovic M (Mar 2005) Memory predecryption: hiding the latency overhead of memory encryption. ACM SIGARCH Comput Archit News 33(1):27–33
3. Suh GE, Clarke D, Gassend B, van Dijk M, Devadas S (2003) AEGIS: architecture for tamper-evident and tamper-resistant processing. In: ICS'03, San Francisco. ACM, New York, 23–26 Jun 2003
4. Suh GE, O'Donnell CW, Sachdev I, Devadas S (2005) Design and implementation of the AEGIS single-chip secure processor using physical random functions. In: Proceedings of the 32nd international symposium on computer architecture (ISCA'05), Madison. IEEE, New York
5. Suh GE, Clarke D, Gassend B, van Dijk M, Devadas S (2003) Efficient memory integrity verification and encryption for secure processors. In: Proceedings of the 36th international symposium on microarchitecture (MICRO-36'03), San Diego
6. Yang J, Gao L, Zhang Y (May 2005) Improving memory encryption performance in secure processors. IEEE T Comput 54(5):630–640
7. Elbaz R, Torres L, Sassatelli G, Guillemin P, Anguille C, Bardouillet M, Buatois C, Rigaud JB (2005) Hardware engines for bus encryption: a survey of existing techniques. In: Proceedings of the design, automation and test in Europe conference and exhibition (DATE'05), Munich. IEEE Computer Society, Los Alamitos

Other Suggested Reading

TCG Home page: http://www.trustedcomputinggroup.org/

Rootkits

Padmaraj M. V. Nair
Department of Computer Science and Engineering,
School of Engineering, Southern Methodist University,
Dallas, TX, USA

Synonyms

Maliciously modified set of administrative tools

Related Concepts

►Malware; ►Malicious Code; ►Trojans; ►Virus; ►Worms

Definition

Rootkits are software packages that set up and maintain a user's environment on a compromised machine.

Background

The term "rootkit" is originated from the UNIX operating system's "root" and the word "kit." Root is the super user in the UNIX environment and kit is a collection malicious tools.

Applications

The technologies and tricks used in rootkits are intended to hide its data and code. A rootkit protects backdoors such as covert channels and local vulnerabilities like distributed denial of service attack, key loggers, passwords sniffers, and agents. Rootkits use a four-step process to be operational. The first step is to find a vulnerable host and then exploit the host. After gaining access to the host, the attacker downloads the rootkit from his site or a dead drop box. The last step is to deploy the kit via an installation script. The installation process disables the history (like shell history in Unix), unpacks components, sets up directory structure, kills system logging activities, modifies system start up files, and hides evidence of the rootkit package. Once installed, rootkit maintains access to the compromised system, attacks other systems, and destroys evidence.

One may classify rootkit software packages into three main categories: binary, kernel, and library kits. Binary kits replace system files with trojan counterpart and deploy files in a hidden directory like "/dev/.lib". They use invisible characters in directory names to make detection and deletion harder. There are tools to adjust time stamps and size of the trojan to the original. Unlike binary kits, Kernal kits hook into system's kernel and modify necessary system calls. These rootkits can make the whole operating system untrustworthy by adding and modifying device drivers and patching the kernel. Library kits replace the standard system libraries and position in such a way that they will be loaded before the system libraries.

Recommended Reading

1. Hoglund G, Butler J (2005) Rootkits: subverting the Windows kernel. Addison-Wesley, Reading
2. Peikari C, Chuvakin A (2004) Security warrior. O'Reilly Media, Sebastopol
3. http://searchmidmarketsecurity.techtarget.com/sDefinition/0,,sid198_gci547279,00.html
4. http://www.rootkit.com/

Routing Anonymity

►Anonymous Routing

RSA Digital Signature Scheme

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

Related Concepts

►Blind Signature; ►Digital Signature Schemes; ►Rabin Digital Signature Scheme; ►RSA Problem; ►RSA Public-Key Encryption

Definition

An *RSA digital signature scheme* is any of several ►digital signature schemes based on the ►RSA Problem.

Theory

In the basic formula for the RSA cryptosystem [17], a digital signature s is computed on a message m according to the equation (►Modular Arithmetic)

$$s \equiv m^d \bmod n \tag{1}$$

where (n, d) is the signer's RSA private key. The signature is verified by recovering the message m with the signer's RSA public key (n, e):

$$m \equiv s^e \bmod n \tag{2}$$

Though the meaning of the value m that is signed with this formula has changed over the years, the basic formula has remained the same since it was introduced in 1977. The purpose of this entry is to survey the main approaches based on that formula, each of which can be referred to as an *RSA digital signature scheme*.

In the definition just given, the value m is itself the message. This is helpful for illustration, but introduces a few challenges:

1. The set of messages that can be signed with the basic formula is limited to the set of integers in the range $[0, n-1]$. A larger value of m could be signed, but the value $m \bmod n$ is all that would be protected (or recovered).
2. Some messages are quite easy to sign, for instance, $m = 0$ always has the signature 0 regardless of the signer's private key, and similarly the signature for $m = 1$ is 1 and for $m = n-1$ it is $n-1$.
3. Every signature value s corresponds to some message m by Eq. 2. It is easy therefore to construct valid message-signature pairs given only the signer's public key by starting with the signature value s (though the messages so obtained might not be anything meaningful).
4. Finally, the signer's willingness to provide signatures on some messages can be exploited by an opponent to obtain signatures on additional messages. As an example, if the opponent wants a signature on a message m but does not want the signer to see m, the opponent can instead ask to have the message $m' \equiv mr^e \bmod n$ signed, where r is a random value. Let s' be the signature on m', and let $s \equiv s'r^{-1} \bmod n$. Then s is the signature on m. Though this property has the benefit of enabling blind signatures, it also presents an avenue for attack. More generally, RSA signatures have a multiplicative property that can be an advantage or a disadvantage depending on the situation: given signatures on messages m_1 and m_2, it is straightforward to determine the signatures on any combination of products of the form $m_1^i m_2^j$.

Due to the various concerns just described, RSA digital signature schemes in practice are typically constructed so that the value m is not the message itself, but rather a *message representative* derived from the message. This approach employs four design principles that address the concerns just noted:

- *Large message space*: The set of messages that can be signed should be as large as possible. For this reason, the value m is typically derived in some way from the result of applying a ►hash function to the actual message to be signed.
- *Nontrivial message representative*: Message representatives such as 0 and 1 should be avoided, or should be very unlikely. In some schemes, this is accomplished by giving m a "random" appearance and in others, just by some padding.
- *Sparse message representative space*: Only a small fraction of values of m should be valid message representatives; this makes it unlikely that a random signature s will correspond to a valid m. For this reason, the value m generally has some structure that can be verified, and which is unlikely to occur at random. Alternatively, if many values of m are valid, then it should be difficult to find a message with a given message representative m.
- *Non-multiplicativity*: Multiplicative relationships between signatures should be avoided. Randomness or padding, and structure, both help in this regard.

One of the earliest and simplest improvements over the basic formula is to define the message representative m as the hash of the message M being signed, $m = Hash(M)$.

This meets the first three design goals: messages of any length can be signed; hash values such as 0 and 1 are very unlikely; and assuming the hash values are sufficiently shorter than the modulus n, only a small fraction of values of m will be valid hash values.

However, multiplicative relationships are still a potential problem. In particular, if the hash output is significantly shorter than the modulus (e.g., 160 bits vs 1,024 bits with today's parameter sizes), then it is possible to attack the signature scheme by methods from ►index calculus. Although the opponent cannot factor the modulus, the attacker can readily factor the much shorter hash values. From a sufficiently large set of signatures, the attacker can thereby solve for the signatures on *all* values m in a ►factor base by index calculus – and from those construct the signature on *any* message M for which $Hash(M)$ is smooth with respect to that factor base (►Smoothness). (This approach, observed in the design of the ►PKCS #1 specification [15] discussed below, is an extension of an early attack on the RSA cryptosystem by Desmedt and Odlyzko [8].) On the other hand, if the hash value is as long as the modulus n, then factoring the hash value is as hard as factoring the modulus, so the attack just described is not a concern. This is the basis for the *Full Domain Hash* (FDH) scheme of Bellare and Rogaway [2]. In fact, FDH turns out to have an additional security benefit, which is that it is possible in the ►random oracle model to obtain a proof that the signature scheme is as difficult to break, as the RSA Problem is to solve. (Recall that the RSA Problem is up to a certain ratio to solve for x such that $y \equiv x^e \bmod n$, given y, n, and e, where the target value y is random.) It is instructive to explore that proof briefly, as it is a good example of the modern design of cryptographic schemes that has resulted from the insights of Bellare and Rogaway and other contemporary researchers.

In the proof, the attacker is assumed to have the ability to do the following:

- Obtain signatures from the actual signer on some number of chosen messages M.
- Evaluate the hash function on some number of chosen messages M.

The underlying hash function is modeled as a random oracle (►Random Oracle Model), meaning that the attacker's probability of success is taken over a *random* choice of hash function. Accordingly, it is not enough that the attack works well for *some* hash function; rather, it must work well, on average, for *any* (theoretical) hash function. This is a strong assumption on the attack, but a reasonable starting point for analysis.

The attacker's goal is to produce a new signature s' on a new message M', and to be able to do so in a reasonable amount of time with high probability.

Suppose now that there is such an attacker, represented by an algorithm A. In the proof, this algorithm A is transformed to a second algorithm B that breaks the ►RSA Problem in a similar amount of time and probability. The "reduction" (►Computational Complexity) from B to A takes advantage of the fact that the hash function is a black box, so the attacker cannot look inside. Accordingly, it is possible to "simulate" both the signer and the hash function so that they appear to be actual ones to the algorithm A, but such that when algorithm A forges the signature, it will in fact be solving an instance of the RSA Problem.

The reduction employs a table that has a signature-hash entry for each message, and goes something like this:

1. When the attacker asks to obtain a signature on a message M, the simulated signer first looks in the table to see if there is an entry for the message. If so, the signature part of that entry is returned. If not, see Step 3 below.
2. When the attacker asks to evaluate the hash function on a message M, the "simulated" hash function likewise first looks and returns the hash part of an entry, if there is one. If not, see Step 3 next.
3. In either case, if there is no entry for the message, then a new one is created. First, a random signature s is generated. Second, a message representative $m \equiv s^e \bmod n$ is computed. Third, the pair (m, s) is entered in the table as the hash-signature pair for the message M. Finally, the hash or signature is returned according to the attacker's request.

This simulation looks just like an actual signer and random ►hash function, because the hash value for a given message is random (since s is random), and the hash-signature pair for each message is internally consistent. Thus, an attacker will be just as successful when interacting with this simulation, as in the actual environment. But notice that the simulation does not need the actual signer's private key. So the simulator, combined with the algorithm A, yields another algorithm B, also independent of the actual signer's private key, which can then be applied to solve the RSA Problem.

In order to produce a forged signature on some message other than by accident, the attacker has to ask for the hash of the message. Otherwise, since the hash value is random, the probability that the signature will match is $1/n$, that is, essentially 0. Because this signature is a forgery and FDH is deterministic, the attacker cannot ask for another signature on the same message. This means that there will be at least one message that the attacker asks to be hashed, but does not ask to be signed.

The one extra message gives the simulator an opportunity to embed an instance of the RSA Problem into the hash value. This is done with a small change to Step 2 above. In particular, at one randomly selected time during the interaction with the attacker, instead of constructing a new entry, the step returns, as the hash value on that message, the target value y for the RSA Problem to be solved. Now, if the attacker also asks for a signature on the same message, the simulation will fail. But if the attacker produces a new signature on that message – a forgery – then the signature will be the solution to the RSA Problem on y!

Algorithm B is thus able to solve the RSA Problem whenever algorithm A produces a forgery, provided that the hash value involved in the forgery is the one selected by the simulator (and Algorithm A uses the hash value, rather than guessing the signature). Suppose algorithm A succeeds with probability ε_A in time T_A, and asks for at most Q messages to be hashed. Then algorithm B succeeds with probability at least $(\varepsilon_A - 1/n)/Q$ in time just slightly more than T_A. Conversely, this means that if there is no algorithm for solving the RSA Problem that succeeds with probability greater than ε_B in time T_B, then there is no algorithm for forging signatures with probability greater than $\varepsilon_B Q + 1/n$ in time T_B.

Because in practice an attacker can run a hash function a very large number of times, the ratio between the probabilities ε_A and ε_B may be quite large. Thus, the security proof itself may not imply as great a minimum difficulty as one might desire for forging FDH signatures, at least for typical parameter sizes. For instance, if one believes that the probability of breaking the RSA problem in a certain amount of time is at most 2^{-80}, and that an attacker can run the hash function 2^{64} times, then the proof only shows that the probability of forgery is at most about 2^{-16}. This does not mean that there is an algorithm that is successful to this extent; it just means that such an algorithm cannot be ruled out by the proof itself. Still, the line of reasoning is better than for "ad hoc" designs, where there is no clear connection between the difficulty of forgery and the RSA Problem.

The reason that the reduction for FDH is so "loose" (i.e., the ratio is so high) is that the simulator is able to embed the RSA Problem into only one hash value. Another scheme by the same authors, the *Probabilistic Signature Scheme* (PSS) [3], overcomes this limitation by introducing a random value (called a *seed*, which plays a role similar to *salt*) into each signature operation. As a result, each signature is independent of previous hash operations, so each hash value can embed a separate instance of the RSA Problem.

In PSS, the probability of success for breaking the RSA Problem is about the same as the probability for forging signatures, which is the best security reduction one can achieve.

(Some improvement in the security reduction for FDH can be obtained by a better proof technique, as Coron has shown [5]. Coron also gives a very careful analysis of the effect of the size of the salt on the reduction in PSS.) In addition to the tight security proof, PSS has a second advantage: It has a simple variant, called PSS-R, that provides *message recovery*. In PSS-R, part or all of the message can be carried in the message representative in addition to the hash value. This is a return to the goal of the original formula, but with stronger security properties.

Applications

The digital signature schemes just described appear in industry standards and are employed in practice to varying degrees.

For the long term, PSS and PSS-R may well prevail as the most common RSA signature schemes, and they are found in newer standards such as ►PKCS #1 v2.1 [16], ISO/IEC 9796-2 [13] and ISO/IEC 14888-2 [14], and IEEE 1363a [11]. However, in the near term, other signature schemes are better established. The most common today in practice is the PKCS #1 v1.5 scheme, introduced in 1991 in the first set of ►Public-Key Cryptography Standards from RSA Laboratories [13]. The scheme has an ad hoc design where the message representative is constructed from a hash value with simple padding. On the one hand, no practical attack has been developed on this scheme, though some specially constructed cases have been shown to have weaknesses (see [7]). On the other hand, no security proof is available either, and it seems unlikely that one would be developed. Thus, like primitives from symmetric cryptography, the security of the scheme depends on resistance against specific attacks rather than a security reduction from a hard problem.

Another scheme today, found more in standard documents than in practice, is the scheme in ANSI X9.31 [1], which is also in ISO/IEC 14888-2. (A variant with message recovery is in ISO/IEC 9796-2 [13].) This scheme also has an ad hoc design with similar security properties to ►PKCS #1 v1.5.

An early scheme with message recovery can be found in ISO/IEC 9796-1 [12]. The scheme is particularly attractive for implementation since no hash function is involved. However, the scheme has turned out to be vulnerable to attack in some cases, as shown by Grieu [10] and in further work with by Coppersmith et al. [4]. The standard has since been withdrawn. (Not all implementations of the standard are affected by the attacks; in particular, implementations where the message being signed and "recovered" is itself a hash value are not affected.) Some of the RSA signature schemes mentioned also have variants based on the ►Rabin cryptosystem; this is true of the schemes in ANSI X9.31 and the various ISO/IEC documents, as well as PSS and PSS-R, though not the PKCS #1 v1.5 scheme. The variants based on the Rabin cryptosystem have the advantage that they allow the public exponent $e = 2$, so that signature verification is very efficient; but on the other hand, some extra steps are required due to the fact that not every message representative m may have a square root modulo n (see ►Rabin Digital Signature Scheme for further discussion.)

A complementary approach to the schemes just described, which is primarily of research interest so far, is to derive the public exponent e itself from the message, where the value m is fixed within the public key. The advantage of this approach, described by Gennaro et al. [9], is that it is possible to obtain a tight security proof in the standard model, where the hash function is only assumed to have a certain "division-intractability" property – it does not need to be modeled as a random oracle. The difficulty of forging a signature can be shown to be closely related to the ►Strong RSA Assumption (refer again ►RSA Problem). (Note though that the initial analysis needed some improvements [6].) A related approach is presented by Vanstone and Qu [18]; in their approach, both e and m may be derived from the message.

Recommended Reading

1. American National Standard X9.31-1998. Digital Signatures Using Reversible Public Key Cryptography for the Financial Services Industry (rDSA)
2. Bellare M, Rogaway P (1993) Random oracles are practical. A paradigm for designing efficient protocols. In: Proceedings of the 1st ACM conference on computer and communications security, Fairfax, November 1993. ACM Press, New York, pp 62–73
3. Bellare M, Rogaway P (1996) The exact security of digital signatures – how to sign with RSA and Rabin. In: Maurer U (ed) Advances in cryptology – EUROCRYPT'96. Lecture notes in computer science, vol 1070. Springer, Berlin, pp 399–416
4. Coppersmith D, Coron J-S, Grieu F, Halevi S, Jutla CS, Naccache D, Stern JP (2008) Cryptanalysis of ISO/IEC 9796-1. J Cryptol 21:27–51
5. Coron J-S (2002) Optimal security proofs for PSS and other signature schemes. In: Knudsen LR (ed) Advances in cryptology – EUROCRYPT 2002. Lecture notes in computer science, vol 2332. Springer, Berlin, pp 272–287
6. Coron J-S, Naccache D (2000) Security analysis of the Gennaro–Halevi–Rabin signature scheme. In: Preneel B (ed) Advances in cryptology – EUROCRYPT 2000. Lecture notes in computer science, vol 1807. Springer, Berlin, pp 91–101
7. Coron J-S, Naccache D, Stern JP (1999) On the security of RSA padding. In: Wiener M (ed) Advances in cryptology – CRYPTO'99. Lecture notes in computer science, vol 1666. Springer, Berlin, pp 1–18
8. Desmedt Y, Odlyzko AM (1986) A chosen text attack on the RSA cryptosystem and some discrete logarithm schemes. In: Williams HC (ed) Advances in cryptology – CRYPTO'85. Lecture notes in computer science, vol 218. Springer, Berlin, pp 516–522
9. Gennaro R, Halevi S, Rabin T (1999) Secure hash-and-sign signatures without the random oracle. In: Stern J (ed) Advances in cryptology – EUROCRYPT'99. Lecture notes in computer science, vol 1592. Springer, Berlin, pp 123–139
10. Grieu F (2000) A chosen messages attack on the ISO/IEC 9796-1 signature scheme. In: Preneel B (ed) Advances in cryptology – EUROCRYPT 2000. Lecture notes in computer science, vol 1807. Springer, Berlin, pp 70–80
11. IEEE Std 1363a:2004. Standard specifications for public key cryptography – amendment 1 – additional techniques
12. ISO/IEC 9796-1:1999. (withdrawn) Information technology – security techniques digital signatures giving message recovery – part 1: mechanisms using redundancy
13. ISO/IEC 9796-2:2002. Information technology – security techniques – digital signatures giving message recovery – part 2: integer factorization based mechanisms
14. ISO/IEC 14888-2:2008. Information technology – security techniques – digital signatures with appendix – part 2: integer factorization based mechanisms
15. PKCS #1 v1.5 (1993) RSA encryption standard. RSA Laboratories
16. PKCS #1 v2.1 (2002) RSA cryptography standard. RSA Laboratories
17. Rivest RL, Shamir A, Adleman LM (1978) A method for obtaining digital signatures and public-key cryptosystems. Commun ACM 21(2):120–126
18. Vanstone SA, Minghua Qu (2000) Digital signature protocol with reduced bandwdith. US Patent 6,097,813, 1 Aug 2000 (Filed 15 May 1997)

RSA Factoring Challenge

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

Related Concepts

►ECC Challenges; ►Integer Factoring; ►MIPS-Year; ►Number Field Sieve for Factoring; ►Quadratic Sieve; ►RSA Problem

Definition

The RSA factoring challenge is a competition run from 1991 to 2007 that awarded cash prizes for the successful factorizations of certain large numbers.

Background

Starting in 1991, RSA Data Security (now part of EMC Corporation) offered a set of "challenges" intended to measure the difficulty of integer factoring. The challenges consisted of a list of 41 *RSA Numbers*, each the product of two prime numbers of approximately equal length, and another, larger list of *Partition Numbers* generated according to a recurrence.

The first five of the RSA Numbers, ranging from 100 to 140 decimal digits (330–463 bits), were factored successfully by 1999 (see [2] for details on the largest of these). An additional 512-bit (155-digit) challenge number was later added in view of the popularity of that key size in practice; it was also factored in 1999 [1].

In addition to the formal challenge numbers, an old challenge number first published in August 1977, renamed "RSA-129," was factored in 1994 [1].

The original challenge was replaced with a second series that ran from 2001 to 2007 with numbers from 576 to 2,048 bits. The numbers in the second series are designated by their length in bits rather than decimal digits.

The ►Quadratic Sieve was employed for the numbers up to RSA-129, and the ►Number Field Sieve for Factoring for the rest.

The work factor in ►MIPS-years, summarized in Table 1, was roughly in line with expectations for these methods as techniques steadily improved. It is noteworthy that the effort for RSA-130, taking advantage of the Number Field Sieve, was less than that for RSA-129. The teams that factored RSA-160 and larger numbers expressed the work factor in terms of specific CPU times rather than converting to MIPS-years. For instance, RSA-200 was reported to have taken the equivalent of 75 processor-years on a 2.2 GHz AMD Opteron processor. This is about twice the effort reported for RSA-640. (Factorizations of numbers in this challenge continue to be reported. As of this writing, the largest number in the challenge to be factored is RSA-768 [5], which took about 2000 processor-years. In addition to the factorizations in Table 1, RSA-170, -180, and -190 have also been factored recently.)

Cash prizes of more than US$50,000 were awarded to the winners over the duration of the contest.

RSA Factoring Challenge. Table 1 Results of the RSA factoring challenge

Number	Bits	Year	Method	MIPS-years
RSA-100	330	1991	QS	7
RSA-110	364	1992	QS	75
RSA-120	397	1993	QS	830
RSA-129	426	1994	QS	5,000
RSA-130	430	1996	NFS	1,000
RSA-140	463	1999	NFS	2,000
RSA-155	512	1999	NFS	8,000
RSA-160	530	2003	NFS	Not given
RSA-576	576	2003	NFS	Not given
RSA-640	640	2007	NFS	Not given
RSA-200	663	2005	NFS	Not given
RSA-768	768	2009	NFS	Not given

Adapted from [4]

QS quadratic sieve, *NFS* number field sieve

Applications

Competitions like the RSA Factoring Challenge and the ►ECC Challenges are helpful in establishing confidence in a cryptosystem as they provide a focal point for demonstrating the effectiveness of algorithms for breaking the cryptosystem.

Recommended Reading

1. Atkins D, Graff M, Lenstra AK, Leyland PC (1995) The magic words are Squeamish Ossifrage. In: Pieprzyk J, Safavi–Naini R (eds) Advances in cryptology – ASIACRYPT'94. Lecture notes in computer science, vol 917. Springer, Berlin, pp 263–277
2. Cavallar S, Dodson B, Lenstra A, Leyland P, Lioen W, Montgomery PL, Murphy B, te Riele H, Zimmermann P (1999) Factorization of RSA-140 using the number field sieve. In: Lam KY, Okamoto E, Xing C (eds) Advances in cryptology – ASIACRYPT'99. Lecture notes in computer science, vol 1716. Springer, Berlin, pp 195–207
3. Cavallar S, Dodson B, Lenstra A, Leyland P, Lioen W, Montgomery PL, Murphy B, te Riele H, Zimmermann P et al (2000) Factorization of a 512-bit RSA modulus. In: Preneel B (ed) Advances in cryptology – EUROCRYPT 2000. Lecture notes in computer science, vol 1807. Springer, Berlin, pp 1–18
4. Contini S (1999) The factorization of RSA-140. RSA Laboratories' Bulletin 10. http://www.rsa.com/rsalabs. Accessed 8 March 1999
5. Kleinjung T, Aoki K, Franke J, Lenstra AK, Thomé E, Bos JW, Gaudry P, Kruppa A, Montgomery PL, Osvik DA, te Riele H, Timofeev A, Zimmermann P (2010) Factorization of a 768-bit RSA modulus. In: Rabin T (ed) Advances in cryptology – CRYPTO 2010. Lecture notes in computer science, vol 6223. Springer, Berlin, pp 333–350

RSA Problem

Ronald L. Rivest[1], Burt Kaliski[2]
[1]Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, USA
[2]Office of the CTO, EMC Corporation, Hopkinton, MA, USA

Related Concepts

►Discrete Logarithm Problem; ►Integer Factoring; ►RSA Digital Signature Scheme; ►RSA Public-Key Encryption; ►Strong RSA Assumption

Definition

Given an RSA public key (n, e) and a ciphertext C, the *RSA Problem* is to find a message M such that

$$C = M^e \, (\bmod \, n) \, .$$

Theory

In ▶RSA public-key encryption [30], Alice encrypts a plaintext M for Bob using Bob's encryption key (n, e) by computing the ciphertext

$$C = M^e \, (\bmod \, n) \, , \tag{1}$$

where n, the *modulus*, is the product of two or more large primes, and e, the *public exponent*, is an (odd) integer $e \geq 3$ that is relatively prime to $\phi(n)$, the *order* of the multiplicative *group* $\mathbf{Z}_n^*$. (Refer ▶Euler's totient function, ▶modular arithmetic for background on these concepts.) Bob, who knows the corresponding RSA decryption key (n, d), can easily decrypt since $de = 1(\bmod \phi(n))$ implies that

$$M = C^d \, (\bmod \, n) \, . \tag{2}$$

An adversary may learn C by eavesdropping and may very well also know Bob's public key; nonetheless, such an adversary should not be able to compute the corresponding plaintext M.

One may formalize the task faced by this adversary as the *RSA Problem*.

The RSA Problem: Given an RSA public key (n, e) and a ciphertext $C = M^e (\bmod \, n)$, to compute M.

To solve the RSA Problem, an adversary, who doesn't know the private key, must nonetheless invert the RSA function.

The *RSA Assumption* is that the RSA Problem is hard to solve when the modulus n is sufficiently large and randomly generated, and the plaintext M (and hence the ciphertext C) is a random integer between 0 and $n - 1$. The assumption is the same as saying that the RSA function is a ▶trapdoor one-way function (the private key is the trapdoor).

The randomness of the plaintext M over the range $[0, n - 1]$ is important in the assumption. If M is known to be from a small space, for instance, then an adversary can solve for M by trying all possible values for M.

The RSA Problem is the basis for the security of RSA public-key encryption as well as ▶RSA digital signature schemes.

See also surveys by Boneh [10] and Katzenbeisser [24].

Relationship to Integer Factoring

The RSA Problem is clearly no harder than ▶integer factoring since an adversary who can factor the modulus n can compute the private key (n, d) from the public key (n, e).

However, it is not clear whether the converse is true, i.e., whether an algorithm for integer factoring can be efficiently constructed from an algorithm for solving the RSA Problem.

Boneh and Venkatesan [9] have given evidence that such a construction is unlikely when the public exponent is very small, such as $e = 3$ or 17. Their result means that the RSA Problem for very small exponents could be easier than integer factoring, but it doesn't imply that the RSA Problem is actually easier, i.e., efficient algorithms are still not known. For larger public exponents, the question of equivalence with integer factoring is still open as of this writing.

Recovering the Private Key

Clearly, if the adversary could compute Bob's private key (n, d) from his public key (n, e), then the adversary could decrypt C using Eq. 2.

However, DeLaurentis [15] and Miller [27] have shown that computing an RSA private key (n, d) from the corresponding RSA encryption key (n, e) is as hard as factoring the modulus n into its prime factors p and q. As already noted, given the factors p and q, it is easy to compute d from e, and conversely, there is a probabilistic ▶polynomial-time algorithm which takes as input n, e, and d, and which factors n into p and q (see also Fact 1 in Boneh [10]). If the modulus n was chosen as the product of two "sufficiently large" randomly-chosen prime numbers p and q, then the problem of factoring n appears to be intractable. Thus, the private exponent d is protected from disclosure by the difficulty of factoring the modulus n.

An adversary might also try to compute d using some method of solving the ▶discrete logarithm problem. For example, an adversary could compute the discrete logarithm of M to the base $M^e (\bmod \, n)$. If d is too small (say, less than 160 bits), then an adversary might be able to recover it by the baby step-giant step method.

Even if d is too large to be recovered by discrete logarithm methods, however, it may still be at risk.

For example, Wiener [33] has shown that if the private exponent is less than $n^{1/4}/3$, an adversary can efficiently compute d given n and e. An improved bound of $n^{0.292}$ has been presented by Boneh and Durfee [8] (▶Wiener, Boneh-Durfee, and May Attacks on the RSA Public Key Cryptosystem).

However, it does appear to be the case that if the RSA parameters were chosen large enough, then the adversary cannot solve the RSA Problem by computing the private RSA exponent of the recipient.

Self-Reducibility

It is conceivable that someone could devise a clever procedure for solving the RSA Problem without factoring the modulus n or determining the private key d. An adversary might, for example, have a procedure that decrypts a small fraction of "weak" ciphertexts. However, the RSA procedure enjoys a certain kind of "self-reducibility," since it is multiplicative:

$$(MR)^e = M^e R^e \pmod n .$$

An adversary can transform a given ciphertext M^e into another one $(MR)^e$ by multiplying it by the encryption R^e of a randomly chosen element R of $\mathbf{Z}_n^*$. Since the result has a chance of being a "weak" ciphertext, it follows that if there is an adversarial procedure A that can decrypt a fraction ε of ciphertexts, then there is another (randomized) adversarial procedure A' that can decrypt all ciphertexts in expected running time that is polynomial in the running time of A, in $1/\varepsilon$, and in $\log n$ (►Polynomial Time) (See Motwani and Raghavan [28, Section 14.4]). Self-reducibility is a double-edged sword. On the one hand, it provides assurance that "all" random ciphertexts are equally hard to invert. This property has been helpful in the security proofs for several public-key encryption and signature schemes based on the RSA Problem. On the other hand, self-reducibility provides an avenue for an adversary to gain information about the decryption of one ciphertext from the decryption of other ciphertexts (see section "Chosen ciphertext attacks") below.

Low Public-Exponent RSA

A user of the RSA cryptosystem may reasonably wish to use a public exponent e that is relatively short: common choices are $e = 3$ or $e = 2^{16} + 1 = 65537$. Using a short public exponent results in faster public-key encryption and faster public-key signature verification. Does this weaken RSA?

If the public exponent is small and the plaintext M is very short, then the RSA function may be easy to invert: in particular, if $M < \sqrt[e]{N}$, then $C = M^e$ over the integers, so M can be recovered as $M = \sqrt[e]{C}$.

Håstad [22] shows that small public exponents can be dangerous when the same plaintext is sent to many different recipients, even if the plaintext is "padded" in various (simple) ways beforehand.

Coppersmith et al. [12] give a powerful "related-messages" attack, which is effective when the public exponent is small, based on the LLL algorithm [25] for ►lattice reduction.

Because of these concerns, small public exponents are sometimes avoided in industry standards and in practice. However, the concerns can also be addressed with appropriate padding schemes (see section "Chosen ciphertext attacks" below), provided they are correctly implemented. For digital signature schemes, small public exponents are generally not an issue.

Strong RSA Assumption

The ►Strong RSA Assumption was introduced by Barić and Pfitzmann [3] and by Fujisaki and Okamoto [18] (see also [13]).

This assumption differs from the RSA Assumption in that the adversary can select the public exponent e. The adversary's task is to compute, given a modulus n and a ciphertext C, *any* plaintext M and (odd) public exponent $e \geq 3$ such that $C = M^e (\bmod\ n)$. This may well be easier than solving the RSA Problem, so the assumption that it is hard is a stronger assumption than the RSA Assumption. The Strong RSA Assumption is the basis for a variety of cryptographic constructions.

Bit-Security of RSA Encryption

It is conceivable that RSA could be "secure" in the sense that the RSA Assumption holds (i.e., RSA is hard to invert), yet still "leak" information in that certain plaintext bits are easy to predict from the ciphertext. Does RSA provide security to individual bits of plaintext?

Goldwasser et al. [21] first studied the bit-security of RSA, showing that an adversary who could reliably extract from a ciphertext the least signficant bit (lsb) of the plaintext would in fact be able to decrypt RSA efficiently (i.e., obtain the entire plaintext efficiently).

This line of research was pursued by other researchers. For example, Vazirani and Vazirani [32] showed that an adversary could still decrypt even with an lsb procedure that was only $0.732 + \varepsilon$ accurate. They also showed that the low-order $\log(\log(n))$ bits of plaintext are $3/4 + \varepsilon$ secure.

Chor and Goldreich [11] improved this result to show that the least-significant bit of RSA plaintext cannot be predicted with probability better than $1/2 + 1/poly(\log(n))$ (under the RSA Assumption). Alexi et al. [1, 2] completed this result to show that the least-significant $\log(\log(n))$ bits are secure in the same sense. (Fischlin and Schnorr [17] provide a simpler and tighter proof of this result.) Håstad and Näslund [23] have shown that *all* of the plaintext bits are well protected by RSA, in the sense that having a non-trivial advantage for predicting any one plaintext bit would enable the adversary to invert RSA completely. (Refer also to ►Hard core bit.)

The results about bit-security of RSA generally involve a *reduction* technique (►Computational Complexity), where an algorithm for solving the RSA Problem is constructed from an algorithm for predicting one (or more) plaintext bits. Like self-reducibility, bit-security is a double-edged sword. This is because the security reductions also provide an avenue of attack on a "leaky" implementation. If an implementation of an RSA decryption operation leaks some bits of the plaintext, then an adversary can potentially solve the RSA Problem for *any* ciphertext just by observing the implementation's behavior on some number of other ciphertexts. Such attacks have been described by Bleichenbacher [7] and by Manger [26].

Chosen Ciphertext Attacks

An adversary may be able to decrypt an RSA ciphertext C by obtaining decryptions (e.g., from the legitimate recipient) of other ciphertexts $C_1, C_2, \ldots, C_k$ (which may or may not be related to C). Such attacks are known as ►chosen ciphertext attacks (CCA1 and CCA2, depending on whether the C_i s are allowed to depend upon C (of course they cannot be equal to C)); see Bellare et al. [4] for details.

(The attacks related to bit-security are a special case of chosen-ciphertext attacks in which the adversary only obtains partial information about the decryption, not the full plaintext.)

Davida [14] first studied chosen ciphertext attacks for RSA utilizing the multiplicative property of RSA.

Desmedt and Odlyzko [16] provided another chosen ciphertext attack based on obtaining the decryption of many small primes.

To defeat chosen ciphertext attacks, researchers have turned to (possibly randomized) "padding" schemes that (reversibly) transform a plaintext before encryption.

One such proposal is Optimal Asymmetric Encryption Padding (►OAEP) [5] which has been proven secure for chosen ciphertext attacks by Fujisaki et al. [19] under the RSA assumption. Other proposals that also avoid chosen ciphertext attacks have better security properties [29, 31]. Refer also ►RSA public-key encryption for related discussion.

Chosen-message attacks on ►digital signature schemes are the analogue to chosen ciphertext attacks on public-key encryption, and various padding schemes have been developed to defeat them as well, such as the Probabilistic Signature Scheme (PSS) of Bellare and Rogaway [6] and the scheme of Gennaro et al. [20]. Refer also ►RSA digital signature scheme.

Conclusions

The RSA Problem is now over 30 years old. The elegant simplicity of the problem has led to numerous observations over the years, some yielding attacks, others avoiding them. Public-key encryption schemes and digital signature schemes have been developed, whose strength is derived fully from the RSA Problem. The remaining open question, still, is how closely the security of the RSA Problem depends on integer factoring, and as with any hard problem in cryptography, whether any methods faster than those currently available for solving the problem will ever be discovered.

Applications

The RSA Problem is the basis for the security of several cryptosystems in practice, as further described in the articles ►RSA public-key encryption and ►RSA digital signature schemes.

Open Problems

The status of various open problems has been described throughout this entry. The most important among them, of course, is whether the RSA Problem is indeed difficult, i.e., to prove a lower bound on the running time of an algorithm that solves the RSA Problem. It would also be theoretically significant to know whether the RSA Problem is as difficult as integer factoring.

Recommended Reading

1. Alexi WB, Chor B, Goldreich O, Schnorr CP (1984) RSA/Rabin bits are $1/2+1/poly(\log(N))$ secure. In: Proceedings of FOCS'84, Singer Island. IEEE, pp 449–457
2. Alexi WB, Chor B, Goldreich O, Schnorr CP (1988) RSA and Rabin functions: certain parts are as hard as the whole. SIAM J Comput 17(2):194–209
3. Barić N, Pfitzmann B (1997) Collision-free accumulators and fail-stop signature schemes without trees. In: Fumy W (ed) Advances in cryptology – EUROCRYPT'97. Lecture notes in computer science, vol 1233. Springer, Berlin, pp 480–494
4. Bellare M, Desai A, Pointcheval D, Rogaway P (1998) Relations among notions of security for public-key encryption. In: Krawczyk H (ed) Advances in cryptology – CRYPTO'98. Lecture notes in computer science, vol 1462. Springer, Berlin, pp 26–45
5. Bellare M, Rogaway P (1996) Optimal asymmetric encryption – how to encrypt with RSA. In: DeSantis A (ed) Advances in cryptology – EUROCRYPT'94. Lecture notes in computer science, vol 950. Springer, Berlin, pp 92–111
6. Bellare M, Rogaway P (1996) The exact security of digital signatures – how to sign with RSA and Rabin. In: Maurer U (ed) Advances in cryptology – EUROCRYPT'96. Lecture notes in computer science, vol 1070. Springer, Berlin, pp 399–416
7. 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
8. Boneh D, Durfee G (2000) Cryptanalysis of RSA with private key d less than $N^{0.292}$. IEEE Trans Inform Theory 46(4):1339–1349
9. Boneh D, Venkatesan R (1988) Breaking RSA may not be equivalent to factoring. In: Nyberg K (ed) Advances in cryptology – EUROCRYPT'98. Lecture notes in computer science, vol 1403. Springer, Berlin, pp 59–71
10. Boneh D (1999) Twenty years of attacks on the RSA cryptosystem. Not Am Math Soc 46(2):203–213
11. Chor B, Goldreich O (1985) RSA/Rabin least significant bits are $1/2 + 1/poly(\log\ n)$ secure. In: Blakley GR, Chaum DC (eds) Advances in cryptology – CRYPTO'84. Lecture notes in computer science, vol 196. Springer, Berlin, pp 303–313
12. Coppersmith D, Franklin M, Patarin J, Reiter M (1996) Low-exponent RSA with related messages. In: Maurer V (ed) Advances in cryptography – EUROCRYPT'96. Lecture notes in computer science, vol 1070. Springer, Berlin, pp 1–9
13. Cramer R, Shoup V (2000) Signature schemes based on the strong RSA assumption. ACM Trans Inform Syst Sec 3(3): 161–185
14. Davida G (1982) Chosen signature cryptanalysis of the RSA (MIT) public key cryptosystem. Technical report TR-CS-82-2, Deptartment of EECS, University of Wisconsin, Milwaukee
15. DeLaurentis JM (1984) A further weakness in the common modulus protocol for the RSA cryptoalgorithm. Cryptologia 8:253–259
16. Desmedt Y, Odlyzko AM (1986) A chosen text attack on the RSA cryptosystem and some discrete logarithm schemes. In: Williams HC (ed) Advances in cryptology – CRYPTO'85. Lecture notes in computer science, vol 218. Springer, Berlin, pp 516–522
17. Fischlin R, Schnorr C-P (2000) Stronger security proofs for RSA and Rabin bits. J Cryptol 13(2):221–244
18. Fujisaki E, Okamoto T (1997) Statistical zero knowledge protocols to prove modular polynomial relations. In: Kaliski BS Jr (ed) Advances in cryptology – CRYPTO'97. Lecture notes in computer science, vol 1294. Springer, Berlin, pp 16–30
19. Fujisaki E, Okamoto T, Pointcheval D, Stern J (2004) RSA-OAEP is secure under the RSA assumption. J Cryptol 17(2): 81–104
20. Gennaro R, Halevi S, Rabin T (1999) Secure hash-and-sign signatures without the random oracle. In: Stern J (ed) Advances in cryptography – EUROCRYPT'99. Lecture notes in computer science, vol 1592. Springer, Berlin, pp 123–139
21. Goldwasser S, Micali S, Tong P (1982) Why and how to establish a private code on a public network. In: Proceedings of the FOCS'82, IEEE, Chicago, pp 134–144
22. Håstad J (1988) Solving simultaneous modular equations of low degree. SIAM J Comput 17:336–341
23. Håstad J, Näslund M (1998) The security of individual RSA bits. In: IEEE symposium on foundations of computer science, Palo Alto, pp 510–521
24. Katzenbeisser S (2001) Recent advances in RSA cryptography. Kluwer, Norwell
25. Lenstra AK, Lenstra HW Jr, Lovász L (1982) Factoring polynomials with rational coefficients. Mathematische Ann 261: 513–534
26. Manger J (2001) A chosen ciphertext attack on RSA Optimal Asymmetric Encryption Padding (OAEP) as standardized in PKCS #1 v2.0. In: Kilian J (ed) Advances in cryptology – CRYPTO 2001. Lecture notes in computer science, vol 2139. Springer, Berlin, pp 260–274
27. Miller GL (1976) Riemann's hypothesis and tests for primality. J Comput Syst Sci 13(3):300–317
28. Motwani R, Raghavan P (1995) Randomized algorithms. Cambridge University Press, Cambridge
29. Okamoto T, Pointcheval D (2001) REACT: rapid enhanced-security asymmetric cryptosystem transform. In: Naccache D (ed) Proceedings cryptographers' track RSA conference (CT-RSA) 2001. Lecture notes in computer science, vol 2020. Springer, Berlin, pp 159–175
30. Rivest RL, Shamir A, Adleman LM (1978) A method for obtaining digital signatures and public-key cryptosystems. Commun ACM 21(2):120–126
31. Shoup V (2001) A proposal for an ISO standard for public key encryption (version 2.1). Manuscript, December 20. http://shoup.net/papers/
32. Vazirani U, Vazirani V (1984) RSA bits are.732 + ε secure. In: Chaum D (ed) Proceedings of the CRYPTO'83. Plenum, New York, pp 369–375
33. Wiener M (1990) Cryptanalysis of short RSA secret exponents. IEEE Trans Inform Theory 36(3):553–558

RSA Public-Key Encryption

David Pointcheval
Computer Science Department, Ecole normale supérieure, Paris, France

Related Concepts

►Integer Factoring; ►OAEP; ►Rabin

Definition

RSA public-key encryption (due to Rivest, Shamir, and Adleman) is an asymmetric encryption scheme based on the RSA trapdoor one-way permutation, and thus related to integer factoring.

Theory

One-Way Permutations

A ►one-way function is a function f that anyone can compute efficiently, however inverting f is hard. Such a primitive is the basis of modern cryptography, and relies on the open problem $\mathcal{P}$ vs. $\mathcal{NP}$ (►computational complexity). As a consequence, any $\mathcal{NP}$-complete problem should lead to such a one-way function candidate. Unfortunately, $\mathcal{NP}$-complete problems are not so convenient for cryptographic applications, because either they are hard to solve for very large instances only, or very few instances are hard but the problem is easy on average. Furthermore, such a primitive is not enough for public-key encryption.

A ▸trapdoor one-way permutation primitive (▸substitutions and permutations) is a permutation f onto a set X that anyone can compute efficiently; however, inverting f is hard unless one is also given some "trapdoor" information. Given the trapdoor information, computing g the inverse of f becomes easy. Naively, a trapdoor one-way permutation defines a simple public-key encryption scheme (▸public key cryptography): The description of f is the public key and the trapdoor, or equivalently the inverse permutation g, is the secret key. As a consequence, in order to send a message $m \in X$ to the owner of the public key f, one computes $c = f(m)$. The recipient is the only one to know the trapdoor, and thus the only one able to compute $m = g(c)$.

Number Theory

A first simple candidate that may come to mind as a one-way function, except $\mathcal{NP}$-complete problems, is integer multiplication: While it is easy to multiply two prime integers p and q to get the product $n = p \cdot q$, there is no easy way to get back p and q from n. Indeed, the product of two integers p and q of similar bit-size k just requires a quadratic amount of time in k. However, the factorization of any integer n, which consists of either writing n as a product of ▸prime numbers $n = \Pi p_i^{v_i}$ – which decomposition is unique up to a permutation of the factors – or just extracting one factor, is much more intricate. Factorization is indeed believed to be a quite difficult problem (▸integer factorization), especially for products of two primes of similar sizes.

Unfortunately, integer multiplication is just one-way. And no trapdoor can make inversion easier. However, some algebraic structures are based on the factorization of an integer n, where some computations are difficult without the factorization of n, but easy with it: In the finite quotient ring $\mathbb{Z}_n = \mathbb{Z}/n\mathbb{Z}$ (▸modular arithmetic), one can easily compute basic operations (equality test, addition, subtraction or multiplication). About inversion, Bézout's theorem gives a theoretical result, while the extended ▸Euclidean algorithm gives the constructive version, since it explicitly computes u and v:

Theorem 1 (Bézout) Let a and n be two integers, then there exist $u, v \in \mathbb{Z}$ such that $au + nv = 1$ if and only if $\gcd(a, n) = 1$.

As a consequence, for any $a \in \mathbb{Z}_n$, a is invertible if and only if a is co-prime to n and, the extended Euclidean algorithm efficiently provides the inverse u. Furthermore, the following corollary comes from the fact that a prime number is co-prime with any positive integer, less than itself:

Corollary 1 *The integer p is a prime $\Leftrightarrow$ the ▸ring $\mathbb{Z}_p$ is a ▸field.*

Therefore, the multiplicative ▸group $\mathbb{Z}_p^*$ of the inverses modulo the prime p contains all the nonzero elements. When n is not a prime, $\mathbb{Z}_n$ is not a field, but the ▸Chinese Remainder Theorem provides the structure of $\mathbb{Z}_n$, with an explicit isomorphism of rings:

Theorem 2 (Chinese Remainder Theorem) Let $n = m_1 m_2$ be a composite integer, where $\gcd(m_1, m_2) = 1$. Then the ring $\mathbb{Z}_n$ is isomorphic to the product ring $\mathbb{Z}_{m_1} \times \mathbb{Z}_{m_2}$.

About the multiplicative group $\mathbb{Z}_n^*$, one gets the following corollary:

Corollary 2 *Let $n = m_1 m_2$ be a composite integer, where* $\gcd(m_1, m_2) = 1$.

$$(\mathbb{Z}_n^*, \times) \simeq (\mathbb{Z}_{m_1}^*, \times) \times (\mathbb{Z}_{m_2}^*, \times).$$

The well-known ▸Euler's Totient Function $\varphi(n)$ is defined by the cardinality of the multiplicative group $\mathbb{Z}_n^*$. Thanks to the Chinese Remainder Theorem, and in the above corollary, this function is weakly multiplicative, which means:

$$\gcd(m_1, m_2) = 1 \Rightarrow \varphi(m_1, m_2) = \varphi(m_1)\varphi(m_2).$$

Since $\varphi(p^v) = p^v - p^{v-1}$ for any prime p and any valuation $v \geq 1$, one can deduce that for any integer n

$$n = \prod_1^{\ell} p_i^{v_i} \Rightarrow \varphi(n) = n \times \prod_1^{\ell} \left(1 - \frac{1}{p_i}\right).$$

Theorem 3 The computation of $\varphi(n)$ is polynomially equivalent to the factorization of n.

Proof It is clear that the factorization of n easily leads to the value of $\varphi(n)$ with the above formula. Furthermore, Miller's algorithm [4] outputs the factorization of any n, given a multiple of $\varphi(n)$. □

Modular Powers and Roots

In any group (denoted multiplicatively), the power to a scalar c can be performed with a linear complexity in the size k of this scalar, using the *square-and-multiply* technique (►exponentiation algorithms):

$$x^c = x^{\sum_{i=0}^{i=k} c_i 2^i} = \prod_{i=0}^{i=k} x^{c_i \times 2^i}$$
$$= \prod_{i=0}^{i=k} x_i^{c_i}, \quad \text{where } x_0 = x \text{ and } x_i = x_{i-1}^2,$$

where $c_0, c_1, \ldots, c_k$ denotes the binary expansion of c. On the other hand, root-extraction (►modular root) does not admit any generic algorithm, unless the order of the group is known. Indeed, the classical Lagrange's theorem provides a solution:

Theorem 4 (Lagrange's Theorem) Let G be any group, and c its cardinality, for any element $x \in G$, $x^c = 1$.

This theorem applied to the particular situation of the multiplicative group $\mathbb{Z}_n^*$ becomes:

Theorem 5 (Euler's Theorem) Let n be any integer, for any element $x \in \mathbb{Z}_n^*$, $x^{\varphi(n)} = 1 \bmod n$.

Therefore, for any integer e relatively prime to $\varphi(n)$, and any $x \in \mathbb{Z}_n^*$, if one takes $d = e^{-1} \bmod \varphi(n)$, which means that there exists an integer $k \in \mathbb{Z}$ such that $ed + k\varphi(n) = 1$ and the values of d (and k) can be easily computed with the extended Euclidean algorithm, then

$$(x^e)^d = x^{ed} = x^{1-k\varphi(n)} = x \cdot (x^{\varphi(n)})^{-k} = x \bmod n.$$

As a consequence, $y = x^d \bmod n$ is the eth root of x in $\mathbb{Z}_n^*$.

As previously seen, the eth power can be easily computed using the *square-and-multiply* method. The above relation allows to easily compute eth roots, by computing dth powers, where $ed = 1 \bmod \varphi(n)$. However, to compute eth roots, one requires to know an integer d such that $ed = 1 \bmod \varphi(n)$. And therefore, $ed - 1$ is a multiple of $\varphi(n)$, which is equivalent to the knowledge of the factorization of n. This provides a trapdoor one-way permutation $f_{n,e}$ whose inverse $g_{n,d}$ requires the knowledge of d, or equivalently the factorization of n:

$$f_{n,e} : x \mapsto x^e \bmod n, \quad g_{n,d} : y \mapsto y^d \bmod n.$$

The RSA Primitive

The RSA Problem

In 1978, Rivest, Shamir, and Adleman [6] defined the following problem.

Definition 2 *The RSA Problem Let $n = pq$ be the product of two large primes and e an integer relatively prime to $\varphi(n)$. For a given $y \in \mathbb{Z}_n^*$, compute $x \in \mathbb{Z}_n^*$ such that $x^e = y \bmod n$.*

It has been shown above that with the factorization of n (the trapdoor), this problem can be easily solved. However, nobody knows whether the factorization is required, but nobody knows how to do without it either, hence the RSA assumption.

Definition 3 *The RSA Assumption For any product of two large primes, $n = pq$, the RSA problem is intractable (presumably as hard as the factorization of n).*

The Plain RSA Cryptosystem

In the RSA cryptosystem, the setup consists of choosing two large prime numbers p and q, and computing the RSA modulus $n = pq$. The public key is n together with an exponent e (relatively prime to $\varphi(n) = (p-1)(q-1)$). The secret key d is defined to be the inverse of e modulo $\varphi(n)$. Encryption and decryption are defined as follows:

$$\mathcal{E}_{n,e}(m) = m^e \bmod n, \quad \mathcal{D}_{n,d}(c) = c^d \bmod n.$$

Security Weaknesses

Unfortunately, encryption in this naive public-key scheme is not secure: The accepted security requirement for an encryption scheme is the so-called ►semantic security (indistinguishability) against an ►adaptive chosen-ciphertext attack [5] or IND–CCA2 for short. But the RSA primitive does not provide by itself an IND–CCA2 secure encryption scheme. Under a slightly stronger assumption than the intractability of the integer factorization, it gives a cryptosystem that is only one-way under ►chosen-plaintext attack – a very weak level of security. Semantic security fails because encryption is deterministic: One can indeed easily note that a deterministic encryption algorithm can never achieve semantic security, since one can check a plaintext candidate by simply re-encrypting it. Even worse, under a CCA2 attack, the attacker can fully decrypt a challenge ciphertext $c = m^e \bmod n$ using the ►homomorphic property of RSA:

$$\mathcal{E}_{n,e}(m_1) \cdot \mathcal{E}_{n,e}(m_2) = \mathcal{E}_{n,e}(m_1 m_2 \bmod n) \bmod n.$$

To decrypt $c = m^e \bmod n$ using a CCA2 attack, one first computes $c' = c \cdot 2^e \bmod n$, then asks for the decryption of $c' \neq c$ and gets $2 \cdot m \bmod n$, finally one can deduce m.

But these are not the only weaknesses of the plain RSA cryptosystem. More subtle attacks have been found.

Multicast Encryption

Håstad [3] showed how to use the Chinese Remainder Theorem to recover the plaintext sent using the plain RSA encryption to several recipients that all have a common and small public exponent. Since the encryption cost is directly related to the size of the public exponent e, it is natural that people want to use the smallest value, that is $e = 3$.

Let us assume that Alice, Bob, and Carole have three distinct RSA moduli n_a, n_b, and n_c, but a common public exponent $e_a = e_b = e_c = e = 3$. If Daniel sends the same message m to each of them, Eve can intercept

$$c_a = m^3 \bmod n_a \quad c_b = m^3 \bmod n_b \quad c_c = m^3 \bmod n_c,$$

and then, granted the ▸Chinese Remainder Theorem, she can compute $c \in \mathbb{Z}_{n_a n_b n_c}$ that satisfies

$$c = c_a \bmod n_a, \quad c = c_b \bmod n_b, \quad c = c_c \bmod n_c.$$

This system is also satisfied by m^3, but the isomorphism between $\mathbb{Z}_{n_a} \times \mathbb{Z}_{n_b} \times \mathbb{Z}_{n_c}$ and $\mathbb{Z}_{n_a n_b n_c}$ says that there should be a unique solution, thus $c = m^3 \bmod n_a n_b n_c$, which equality holds in $\mathbb{Z}$ since $m^3 < n_a n_b n_c$. An easy third root in $\mathbb{Z}$ leads back to m.

Small Exponents

As already noticed, people may be interested in using small exponents, either a small encryption exponent to speed up the encryption process, or a small decryption exponent to speed up the decryption process.

A short encryption exponent may be dangerous, as already remarked above, but even in a single-user environment if the message to be encrypted is short too: Let us assume that one uses a 2048-bit modulus n for a quite secure application (such as electronic transactions), with public exponent $e = 3$. Then, one uses this cryptosystem to encrypt a short message m (a transaction, a credit card number, etc.) over less than 64 characters (i.e., less than 512 bits). The ciphertext is $c = m^3 \bmod n$, but even in $\mathbb{Z}$ since m^3 is over 1,576 bits only which is less than n: the modular reduction does not apply – it may be faster! But on the other hand, to recover m from c, one just has to compute a third root in $\mathbb{Z}$, which does not require any factorization.

One may think about a short decryption exponent, since the decryption process is often performed by a low-power device (i.e., smart card). But let us assume that $n = pq$, with $p < q < 2p$, e is the encryption exponent such that the decryption exponent d is less than $n^{1/4}/3$, and $ed = 1 + k\varphi(n)$:

$$\left|\frac{e}{n} - \frac{k}{d}\right| = \left|\frac{ed - kn}{nd}\right| = \left|\frac{1 - k(p+q-1)}{nd}\right| \\ \leq \frac{k(p+q)}{nd} \leq \frac{3k}{d\sqrt{n}}.$$

Since $e \leq \phi(n)$ and $d \leq n^{1/4}/3$, necessarily $3k \leq n^{1/4}$, and $1/n^{1/4} \leq 1/3d$. As a consequence, the above difference is upper-bounded by $1/2d^2$. Using the continued fractions result, this bound says that k/d is one of the *convergents* of the *continued fraction* (▸integer factoring) expansion of e/n: In polynomial time, one gets d [2, 7].

Applications

The RSA function is a quite nice primitive for cryptographic purposes: It provides not only a public-key encryption scheme, but also signature (▸RSA digital signature scheme). However, the basic primitive cannot be used directly because of the very strong algebraic structure. As a consequence, practical RSA-based cryptosystems randomly pad the plaintext prior to encryption. This randomizes the ciphertext and eliminates the homomorphic property. Paddings have been proposed that provably rule out any attack, under the sole RSA assumption (▸OAEP) [1].

Recommended Reading

1. Bellare M, Rogaway P (1995) Optimal asymmetric encryption–how to encrypt with RSA. In: De Santi A (ed) Advances in cryptology–EUROCRYPT'94. Lecture notes in computer science, vol 950. Springer, Berlin, pp 92–111
2. Boneh D, Durfee G (2000) Cryptanalysis of RSA with private key *d* less than N0:292. IEEE Trans Inform Theory 46(4):1339–1349
3. Håstad J (1988) Solving simultaneous modular equations of low degree. SIAM J Comput 17:336–341
4. Miller G (1976) Riemann's hypothesis and tests for primality. J Comput Syst Sci 13:300–317
5. Rackoff C, Simon DR (1992) Non-interactive 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, Berlin, pp 433–444
6. Rivest R, Shamir A, Adleman L (1978) A method for obtaining digital signatures and public key cryptosystems. Commun ACM 21(2):120–126
7. Wiener M (1990) Cryptanalysis of short RSA secret exponents. IEEE Trans Inform Theory 36(3):553–558

Rule-Based Access Control

▸Mandatory Access Control

Run

Tor Helleseth
The Selmer Center, Department of Informatics, University of Bergen, Bergen, Norway

Related Concepts

▸Gap; ▸Golomb's Randomness Postulates; ▸Maximal-Length Linear Sequences; ▸Pseudo-Noise Sequences (PN-sequences); ▸Sequences

Definition
A run in a binary sequence is a set of consecutive 0s or 1s. A run of 0s is often denoted a 0-run or a ►gap and a run of 1s is often denoted a 1-run or a *block*. A gap of length k is a set of k consecutive 0s flanked by 1s. A block of length k is a set of k consecutive 1s flanked by 0s. A run of length k is a gap of length k or a block of length k.

Applications
The number of runs in a sequence is important in ►Golomb's randomness postulates [1] to evaluate the randomness of a sequence.

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

Running-Key

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

Synonyms
Keystream

Related Concepts
►Stream Cipher; ►Symmetric Cryptography

Definition
In a ►stream cipher, the *running-key*, also called the *keystream*, is the sequence which is combined, digit by digit, to the plaintext sequence for obtaining the ciphertext sequence. The running key is generated by a finite-state automaton called the *running-key generator* or the *keystream generator*.

Runtime Analysis

►Dynamic Analysis

Run-Time Malware Analysis

►Dynamic Malware Analysis