Referentiel général de sécurité

Patrick Lacharme

ENSICAEN

2010-2011

Plan

Référentiel Général de Sécurité (RGS)

Document realized by ANSSI :

Version 1.0 (may 2010), available : `www.ssi.gouv.fr/rgs`

Contents :

Framework for SSI management and security rules, used possibly in relation with other tools (Ebios, ISO 2700x, CC,...).

Global document with 17 annexes (RGS_A_1,...,RGS_A_14 and RGS_B_1, RGS_B_2, RGS_B_3).
Cryptography is widely developped, as a security primitive.

Security objectives :

Confidentiality, integrity, availability and authentication.

Plan

Symmetric cryptography

A **secret key** (or symmetric key) is a key used in a secret-key cryptosystem (or symmetric cryptosystem).

The key must remain secret and the security of such algorithms is related on the protection (and the confidentiality) of the key.

Symmetric cryptosystems are generally fast and are used as primitive for the construction of various cryptographic protocols.

Two types of symmetric primitives :

1. Symmetric encryption schemes used for confidentiality.
2. Message authentication algorithms used for integrity.

Symmetric encryption schemes and size of keys

The confidentiality of data with various size and throughput must be ensured (PIN code, message, data files, secret keys).

Two types of symmetric encryption algorithms :

1. **Block ciphers.**
2. **Stream ciphers.**

The minimum size of keys is related to the complexity of the exhaustive search on the key space (**brute force attacks**).

Rule_SymKey : the minimum size of symmetric keys is 100 bits until 2020 and **128 bits** after 2020.

Examples of sizes of keys :
DES : 56 bits, 3-DES : 112 bits, AES : 128, 192 and 256 bits.

Complexity examples for comparison

2^{30}	number of operations in one second (1 billion of operations by seconds : 1 GHz)
2^{46}	number of operations in one day (1 billion of operations by seconds : 1 GHz)
2^{55}	number of operations in one years (1 billion of operations by seconds : 1 GHz)
2^{90}	number of operations in 15 billions years (1 billion of operations by seconds : 1 GHz)
2^{256}	number of electrons in universe.

Block ciphers

A block cipher is a symmetric key algorithm processing data by **block of fixed length**.

A block cipher maps n-bits plaintext blocks to n-bits ciphertext blocks (n is the block size).

A plaintext of length greater than the size of block is encrypted under a single key, depending to the **mode of operation.**

For n-bits plaintext P_n and n-bits ciphertext C_n, with a fixed key k, the **encryption** map $E_k : P_n \to C_n$ is a **bijection** and is noted $E_k(x) = c$.

The inverse map is the **decryption** function $D_k : C_n \to P_n$ is defined by $D_k(c) = E_k^{-1}(c) = x$.

Examples : from DES to AES

DES, based on Lucifer and published by IBM in 1975, adopted by NIST (FIPS 46-1 and 46-2) in 1977.

3-DES, adopted by NIST (FIPS 46-3) in 1998.

IDEA (Massey and Lai, 1991).

AES (**A**dvanced **E**ncryption **S**tandard) finalists :

Rijndael (Daemen, Rijmen, 1998).

Serpent (Anderson, Biham and Knudsen, 1998).

Twofish (Schneier, Kelsey, ..., 1998).

Rijndael algorithm is proposed by NIST for AES in 2000, and is published in FIPS-197 (2001).

Block size in block cipher

The two elementary characteristics of a block cipher are the size of the secret key and the **size of blocks.**

The minimum size of block is related to dictionary attacks and security of mechanisms, as message authentication, using the block ciphers as primitives.

Rule_BlockSym : the minimum size of blocks for block cipher is 64 bits until 2020 and **128 bits** after 2020.

Examples of size of blocks :
DES : 64 bits, 3-DES : 64 bits and AES : 128 bits.

Cryptanalysis of block ciphers

Two examples of attacks on block ciphers :

1. The **differential cryptanalysis** (Biham, Shamir, 1991).
2. The **linear cryptanalysis** (Matsui, 1993).

The security of a block cipher against these attacks depends to the properties of the **boolean functions** used in the design of block ciphers (also called S-boxes).

Example : S-boxes of DES were optimized against differential attacks and S-boxes of AES are optimized against both attacks.

DES algorithm was not optimized against the linear cryptonalysis, but the attack has no practical implication, because too many pair of plaintext/ciphertext are required.

Block cipher

An attack is characterized by three parameters : N_{op} (the number of operations -encryption or decryption), N_{block} (the number of block ciphered) and N_{mem} (the necessary memory).

Rule_AlgoBlock : no attacks with less than $N_{op} = 2^{100}$ operations must be known for a encryption algorithm until 2020 and less than $N_{op} = 2^{128}$ operation after 2020.

Remark : N_{block} and N_{mem} are not used in the previous rule, but must be also taken into account.

Recommendation_AlgoBlock : the block cipher must be widely studied by the academic community.

Conformity : the **AES** mechanism is conform. The 3-DES algorithm is not recommended (size of keys and blocks).

Block cipher modes of operations

Several mode of operation are specified by NIST in SP 800-38 or in FIPS 81 (1980) :

- ECB (*Electronic code book*).
- CBC (*Cipher block chaining*, 1976).
- CFB (*Cipher feedback*).
- OFB (*Output feedback*).

The message m is broken in blocks of *block_size* bits : $m_1 m_2 \ldots m_n$, where the last block m_n is completed with a **padding** of one 1 and possibly several 0.

Recommendation_ModeChiff1 : a non-deterministic encryption mode is recommended (avoiding that the encryption of two identical messages provides the same ciphertext).

Random initial values (IV) are used for this purpose.

CBC mode for encryption

$c_0 = IV$ and $c_i = E_k(m_i \oplus c_{i-1})$.

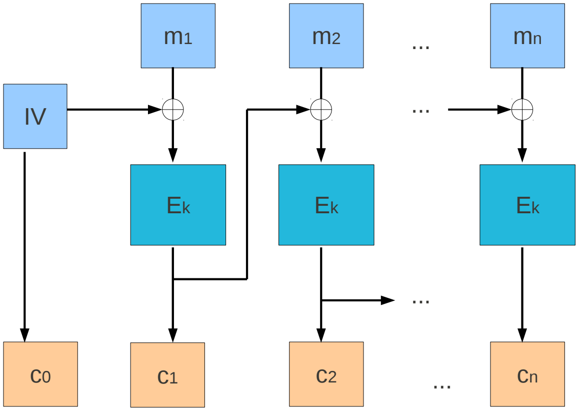

FIGURE: Encryption of message

CBC mode for decryption

$m_i = D_k(c_i) \oplus c_{i-1}$ and $c_0 = IV$.

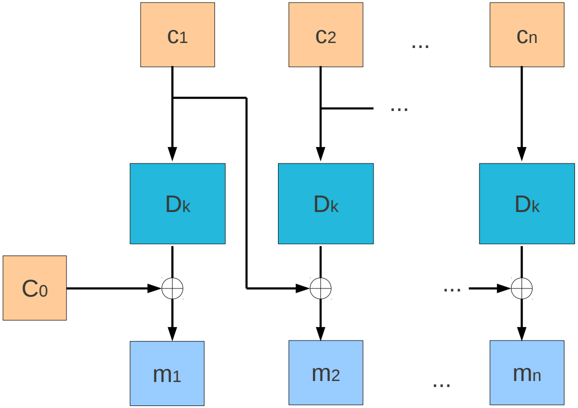

FIGURE: Decryption of message

CFB mode for encryption

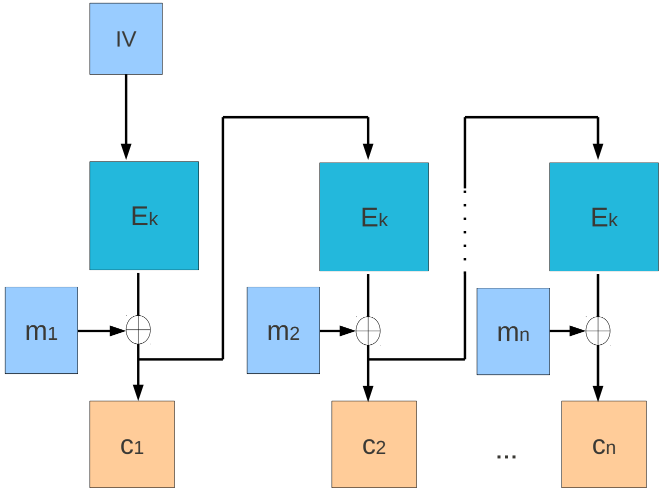

FIGURE: CFB encryption

Block cipher modes of operations

Rule_ModeChiff : no attacks with less than $2^{n/2}$ calls of the encryption primitive must be known (*n* is the size of block).

Recommendation_ModeChiff2 : an integrity mechanism is recommended in addition to an encryption mechanism

Recommendation_ModeChiff3 : the mode of operation must dispose of a security proof.

Conformity : the **CBC** mode of operation (with AES and random IV), using at most $2^{n/2}$ blocks of plaintext, is conform.

Remark : for encryption using block of 64 bits (e.g. 3-DES), the limit for the number of plaintext is approximately 32 Go.

Stream ciphers

Stream ciphers is very efficient for encryption, using on a pseudo-random generator for the generation of a *keystream*.

Encryption is performed with the XOR between the key stream and the plaintext, bit by bit.

Rule_Stream : no attacks with less than 2^{100} operations must be known for a stream cipher algorithm until 2020 and less than $N_{op} = 2^{128}$ operation after 2020.

Remark : N_{mem} and N_{block} must also be taken into account.

Recommendation_Stream1 : the stream cipher must be widely studied by the academic community.

Recommendation_Stream2 : block ciphers in stream mode are preferable to dedicated stream ciphers.

The estream project

The estream project is developped by the **EU Ecrypt network**, in order to propose new stream cipher families (2004).

Finalist (dec. 2008) are the following stream ciphers :

Software profile :

1. HC-128 (Wu)
2. Rabbit (Boesgaard, Vesterager, Christensen and Zenner)
3. Salsa 20/12 (Bernstein)
4. Sosemanuk (12 persons)

Hardware profile :

1. Grain (Hell, Johansson and Meier)
2. Mickey (Babbage and Dodd)
3. Trivium (De Cannière and Preneel)

Example : the A5/1 algorithm used in GSM

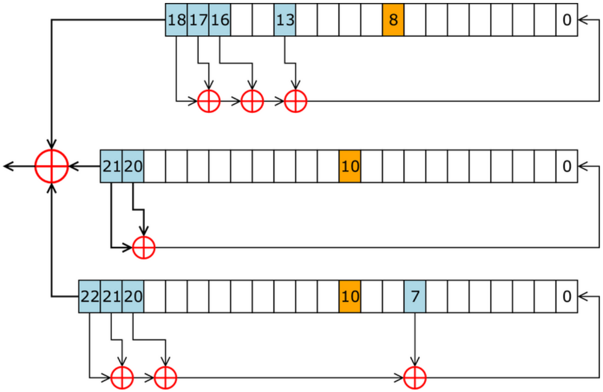

FIGURE: LFSR based A5/1 encryption algorithm

Cryptanalysis : Biryukov, Shamir and Wagner (2000), Biham and Dunkelman (2000), Ekdahl and Johansson (2003).

The A5/2 algorithm is even more vulnerable !

Example : the A5/3 algorithm (Kasumi)

A5/3 uses the block cipher Kasumi, based on the MISTY encryption is the encryption mechanism in GSM.

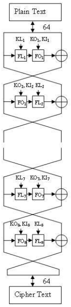

Fig 5. KASUMI Block Cipher

FIGURE: Kasumi algorithm

Cryptanalysis : 2^{56} pairs of plaintext/ciphertext and 2^{76} operations (Biham, Dunkelman and Keller, 2005) and applicable related-key attack (BDK 2010).

Integrity mechansim

Integrity mechanisms are generally used (and recommended) in complement to confidentiality mechanisms (as block cipher).

Rule_IntegSym1 : a block cipher or a hash functions used in an integrity mechanism must be conform.

Rule_IntegSym2 : no attacks with less than $2^{n/2}$ uses of the primitive must be known (n is the output size of the primitive).

Recommendation_IntegSym : an integrity mechanism with a security proof is suitable.

Message authentication codes

A message authentication code (MAC) is a secret-key cryposystem used as an integrity mechanism.

A MAC is sometimes considered as a keyed hash function.

MAC process for integrity control :

1. The sender computes $mac\_1 = MAC\_k(m)$ and sends it with the message m.
2. The receiver computes $mac\_2 = MAC\_k(m)$ and ckecks if $mac\_1 = mac\_2$.

Conformity : the **CBC-MAC** *retail* using the AES encryption and two distinct keys k and k' is conform.

The CBC-MAC using the DES encryption and two distinct keys is not conform.

The CBC-MAC algorithm

The CBC-MAC is a NIST standard (FIPS PUB 113, 1985, NIST SP 800-38B, 2005).

The message m is broken in $m_0 m_1 \ldots m_n$, where the last block m_n is completed with a **padding** of one 1 and several 0.

Conversselly to a block cipher in CBC mode, there is no initialisation vector in the CBC-MAC algorithm.

Algorithm :

let k and k' be the secret keys used by a block cipher E_k.

1. For $i = 1$ to $n - 1$: $c_{i+1} = E_k(c_i \oplus m_i)$.
2. $MAC = E_{k'}(c_n)$

The CBC-MAC description

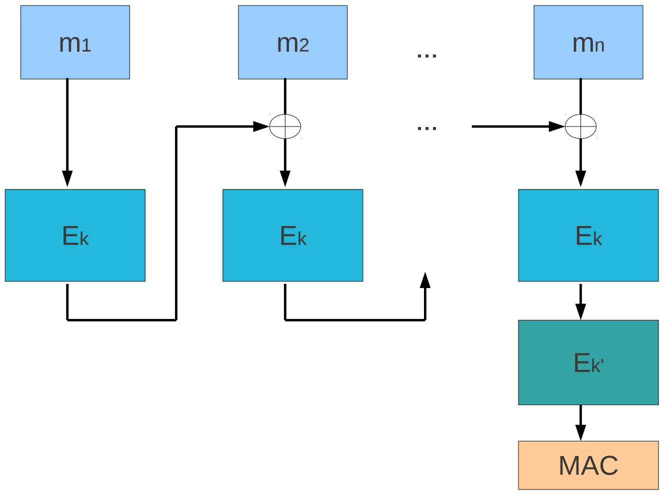

FIGURE: CBC-MAC algorithm

Plan

Hash functions (presentation)

MD4 (Rivest, 1990)

MD5 (Rivest, 1992).

Collisions on **MD4** (Dobbertin, 1995).

Sha-1 (NIST, 1995, defined in FIPS 180-2)

Ripemd-160 (Dobbertin, Bosselaers and Preneel, 1996).

SHA-2 or **Sha-256** (NIST, 2001, defined in FIPS 180-2).

Collisions on **MD5** (Dobbertin, 1996 and Wang, 2004).

Collisions on **Ripemd-160** and **Sha-1** (Wang, 2004 and 2005).

Several collision attacks on **Sha-1** since 2004.

Sha-3 (NIST, 2012).

Requirements for cryptographic hash functions

Let $h : X \rightarrow Y$ be a hash function. Following properties are required for a cryptographic use :

1. **Preimage resistance** (one-way property) :

   Given $y \in Y$, it should be computationally infeasible to find a value $x \in X$ such that $y = h(x)$.

2. **Second preimage resistance** (weak collision resistance) :

   Given $x_1 \in X$, it should be computationally infeasible to find a value $x \in X$ such that $x \neq x_1$ and $h(x) = h(x_1)$.

3. **Collision resistance** (strong collision resistance) :

   It should be computationally infeasible to find two values $x_1 \in X$ and $x_2 \in X$ such that $x_1 \neq x_2$ and $h(x_1) = h(x_2)$.

Attacks on cryptographic hash functions

Let h be a hash function. Hash functions take input with arbitrary length and fixed length output.

Let n be the size of the output.

1. **Preimage resistance :**

   It involves no attacks with complexity less than 2^{n-1}.

2. **Second preimage resistance** :

   It involves no attacks with complexity less than 2^{n-1}.

3. **Collision resistance :**

   It involves no attacks with complexity less than $2^{n/2}$ (**birthday paradox**).

The Merkle-Damgard construction

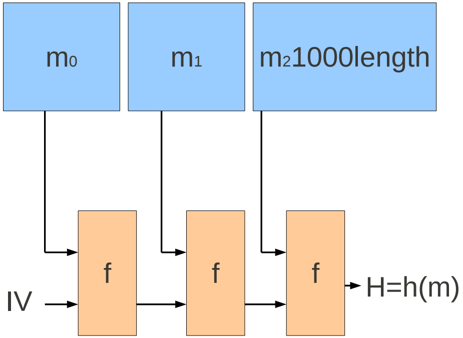

FIGURE: Merkle-Damgard construction

Recommendation_Hash : a hash function with attacks on the compression function *f* is not suitable.

Hash functions (rules and conformity)

Rule_Hash1 : the minimum size of the hash output is 200 bits until 2020 and **256 bits** after 2020.

Rule_Hash2 : the best known attack for finding collision must use $2^{n/2}$ computations, where n is the size of the hash output.

Conformity : the hash function **SHA-256 is conform**.

Non-conformity : the hash function **SHA-1**, **is not conform**.

There exists a collision attack on Sha-1 with complexity $2^{63} < 2^{80}$, and the size of output is 160 bits.

There are no preimage attacks on Sha-1.

Sha-3 standard

NIST hash function competition formally announced on november 2007.

Proclamantion of the winner and publication of the new standard in 2012.

Submission in oct. 2008, acceptation to round 2 in july 2009 and acceptation to the final in august 2010.

The five finalists :

1. Blake (Aumasson, Henzen, Meier, Phan)
2. Grøstl (Knudsen et al.)
3. JH (Wu)
4. Keccak (Bertoni, Daemen, Peeters and Assche)
5. Skein (Schneier et al.)

Example of application : hashcash (1)

HashCash is a mechanism designed to limit spam and denial of service (Back, 1997).

`http://hashcash.org/`

Principle :

The sender must add a stamp (to the header of an email) verifying a properties such that the generation of this stamp take a certain quantity of time (typically 1 second).

The verification of this stamp requires a negligible computation by the receiver.

Example of application : hashcash (2)

Technical details :

For example the header must contains a number *n* such that the first 20 bits of the following hash computation

H(recipient's email address, the date and time, n)

are zeros (H is a hash function such that SHA).

Sender side :

The sender needs to compute 2^{20} hash function before to find a valid *n* (about 1 second with a 1 GHz computer).

Receiver side :

The receiver checks the date and time and computes the hash value before to acept the mail.

Plan

Introduction to asymmetric cryptography

Merkle (1975) : first public key cryptosystem.

Diffie and **Hellman** : key exchange protocol (1976).

Rivest, **Shamir** and **Adleman** : RSA cryptosystem (1978).

Kohnfelder : CA principle, used in PKI (1978).

Public key cryptosystem :

A **private key** is a part of a key used in a asymmetric cryptosystem, and associated to a **public key**.

Security requirements on key management :

Authenticity : cryptographic keys needs **integrity** and must be correctly associated to an entity (**authentication**).

Confidentiality of cryptographic keys is naturally the base for the security of the cryptosystem.

Availability of the cryptograhic keys is necessary for the cryptographic operation.

Rule_KeyMan1 : a key must be used for **one** use (e.g. encryption and signature or encryption and message authentication must use two different keys).

Rule_KeyMan2 : the **key generation** must use a random generator comform and in a trusted environment.

Key derivation

Derivation of several keys from a secret value (**master key**) and a identity element (or a password) uses a pseudo random function, called here **key derivation functions (KDF).**

NIST provides recommendations on key derivation functions in SP 800-89 and proposes a message authentication code (MAC) as pseudo random functions (PRF) :

1. KDF in counter mode
2. KDF in feedback mode
3. KDF in double-pipeline iteration mode

Other key derivation schemes need to be deterministic. In this case, a hash function can be used.

Rule_KeyMan3 : Keys derivated must be generated using a mechanism with adapted security level.

Example of key derivation function : KDF1

KDF1 defined in ISO 18033-2, corresponds to MGF1 (mask generation function) of IEEE P1363a and PKC#1 v 2.1.

KDF1 process (principle) :

Let K be a shared secret (master key), e an identity element (*context*), l a *label* element (if the keying material is derived for different purposes) and PRF a pseudo random function :

For $i = 1$ to n (depending to the length of the final key and the size of the output of the PRF) :

1. $K(i) = PRF(K||i||e||l)$
2. result(i)=result($i-1$)$||K(i)$.

The PRF is a MAC function.

KDF in counter mode (KDF1)

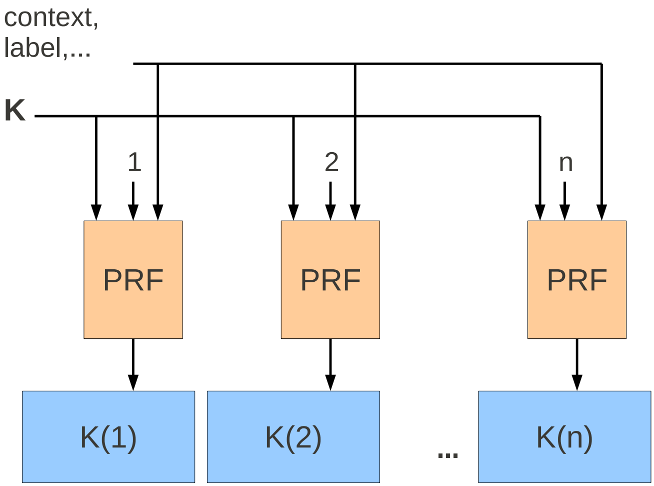

FIGURE: KDF in counter mode

Key hierarchy

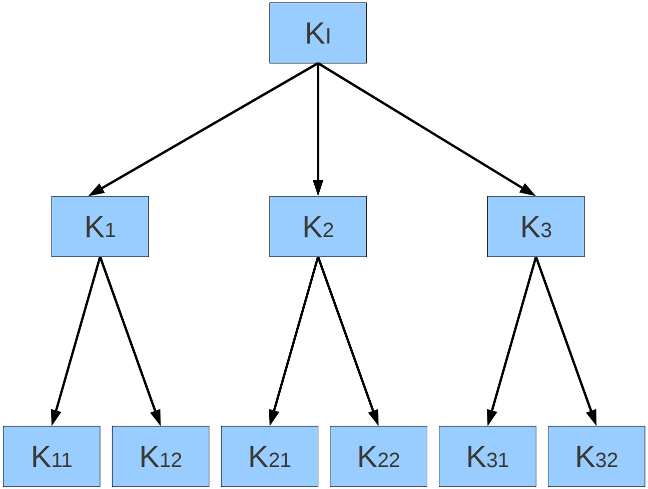

FIGURE: Key hierarchy

Password based KDF (key strengthening)

Password-**B**ased **K**ey **D**erivation **F**unctions.

PBKDF1 and PBKDF2 are defined in PKCS#5 v1 and are used with a password. They use a random *salt* s, in order to provide a resistance against dictionary attacks.

The length of the output key is limited to the length of the hash function. Because the keyspace for passwords is smaller, an iteration counter c is used (typically $c \geq 1000$).

Is it enough ? NO !

Algorithm of PBKDF1 :

Let H be a hash function and p a password :

1. $T_1 = H(P||s)$.
2. For $i = 2$ to c : $T_i = H(T_{i-1})$.
3. Output the firts bytes of T_c as the key.

Discrete logarithm problem (DLP)

The DLP in a cylcic group :

Let G be a cyclic group of order n with generator g. Let A be an element of G. The discrete logarithm problem consists to retrieve a (with $0 \leq a < n$) such that $g^a = A$.

The DLP in a finite field $\mathbf{F}_q$:

Let $\mathbf{F}_q$ be a finite field and g a primitive roots of $\mathbf{F}_q$. Let $A \in \mathbf{F}_q^*$. The discrete logarithm problem consists to retrieve a (with $0 \leq a < q - 1$) such that $g^a = A$.

The DLP in elliptic curve :

Let E be an elliptic curve where the set of points is a group of order n and G, A two points of E. The DLP consists to retrieve a (with $0 \leq a < n$) such that $a.G = A$.

Algorithms for the discrete logarithm problem

Generic methods :
Baby step, giant step (Shanks, 1972).
ρ-method (Pollard, 1978).

Pohlig-Hellman method (Pohlig and Hellman, 1978) :

Pohllig-Hellman method means that the discrete logarithm on a group of order *n* is retrieved from several discrete logarithms on groups of order dividing *n*.

Adleman method (Adleman, 1979) :

This method is sub-exponential, but is only applied for finite fields, and not for a generic group.

Discrete logarithm on $\mathbf{F}_p$

$\mathbf{F}_p$ (or $GF(p)$) denotes the finite field with p elements, where p is a prime number called modulus.

Records for DLP on $\mathbf{F}_p$:

Joux and Lercier : 298 bits (1998), 331 bits (2000), 397 bits (2001) and 432 bits (2005).
Bahr, Franke and Kleinjung : 532 bits (2007).

Rule_DLP1 : the minimum size of the modulus is 2048 bits until 2020 and 4096 bits after 2020.

Rule_DLP2 : The order of sub-groups must be a multiple of a prime number of size greater than 200 bits.

Remark : the complexity of the discrete logarithm problem seems similar to the factorization.

Discrete logarithm on $\mathbf{F}_{2^n}$

$\mathbf{F}_{2^n}$ (or $GF(2^n)$) denotes the finite field with 2^n elements, where n is an integer.

Records for DLP on $\mathbf{F}_{2^n}$:

Gordon and McCurley : 401 bits (1992).
Joux and Lercier : 521 bits (2001) and 613 bits (2005).
Thomé and Bahr : 607 bits (2002).

Rule_DLP3 : the minimum size of n is 2048 bits until 2020 and 4096 bits after 2020.

Recommendation : the discrete logarithm problem on $\mathbf{F}_{2^n}$ **is not recommended**.

The discrete logarithm problem on $\mathbf{F}_{2^n}$ is hard, but more easy than the problem on $\mathbf{F}_p$.

Discrete logarithm in elliptic curves

Rule_EC1 : For a use of elliptic curves on $\mathbf{F}_p$ (p prime), the minimum size of the modulus p is 200 bits until 2020 and 256 bits after 2020.

Conformity : The elliptic curves defined on $\mathbf{F}_p$ P-256, P-384 and P-521 defined in FIPS 186-2 are conform.

Rule_EC2 : For a use of elliptic curves on $\mathbf{F}_{2^n}$, n must be a prime number.

Conformity : The elliptic curves defined on $\mathbf{F}_{2^n}$ B-283, B-409 and P-571 defined in FIPS 186-2 are conform.

Remark FIPS 186-3 recommendation on signature schemes (and more precisely ECDSA) is more recent (2009).

Challenges on discrete logarithm on eliptic curve

List of challenge from certicom company (`www.certicom.com`).

Challenge	Number of operations
ECCp-79	2^{40}
ECCp-89	2^{44}
ECCp-109	2^{54}
ECCp-131	Not solved
ECC2-79	2^{40}
ECC2-89	2^{4}
ECC2-109	2^{54}
ECC2-131	Not solved
ECC2K-108	2^{51}
ECC2K-130	Not solved

Diffie-Hellman (DH) protocol

Rule_KeyMan4 : the **key exchange** algorithm must use cryptographic mechanisms conform to this referential.

Let G be the multiplicative group $\mathbf{F}_p^*$ (with p prime) and g a primitive element. These data are public data.

1. Alice generates a random element a, computes $g^a \bmod p$ and sends it to Bob.
2. Bob generates a random element b, computes $g^b \bmod p$ and sends it to Alice.
3. Alice and Bob computes respectively $(g^a)^b \bmod p$ and $(g^b)^a \bmod p$.

This anonymous (non-authenticated) key agreement protocol is **vulnerable to *man in the middle attacks***.

The **ECDH protocol** is described in NIST SP 800-56A.

Station-to-Station protocol (STS)

STS key agreement protocol (O'Higgins, Diffie, Strawczynski, do Hoog, 1987) provides mutual authentication.

Possibility to use this protocol with elliptic curves and with PKI.

1. Alice generates a random number a, computes and sends g^a to Bob.
2. Bob generates a random number b, computes g^b and the shared key $K = (g^a)^b \bmod p$.
3. Bob concatenes (g^b, g^a), signs them with his private key, encrypts with K and sends the ciphertext with g^b to Alice.
4. Alice computes $K = (g^b)^a \bmod p$, decrypts the ciphertext and verifies the signature.
5. Alice concatenes (g^a, g^b), signs them with his private key, then encrypt with K. She sends the ciphertext to Bob.
6. Bob decrypts the ciphertext and verifies the signature.

Plan

Factorization methods

The product of two prime numbers p and q is called modulus.

Fermat method
This method means that the difference $p - q$ must be great.

p-1 method for factorization (Pollard, 1974).

This method means that there must exist a great prime factor in the decomposition of $p - 1$.

Quadratic sieve or NPQS (Pomerance, 1982)

Factorization of RSA-399 (1993) and RSA-429 (1994).

Algebraic sieve or NFS (Pollard, 1988)

Factorization of RSA-432 (1996), ... , RSA-663 (2005).

Factorization recommendations

Rule_FAC1 : the minimum size of the modulus is 2048 bits until 2020 and 4096 bits after 2020.

(Non) recommendation :
No modulus of 1024 bits have been officially factorized today, but a size of 1024 bits for the modulus is not recommended.

Other recommendations :
The size of p and q must be roughly equal and the difference $p - q$ must be great.
There must exist a great prime factor in the decomposition of $p - 1$ and $q - 1$.

RSA cryptosystem

The RSA cryptosystem is based on factorization problem and uses a public exponent *e* and a private exponent *d*.

Rule_FAC2 : the size of secret exponents must be the same as the modulus.

Rule_FAC3 : for encryption application the size of the public exponent must be greater than $2^{16} = 65536$.

Recommendation : for any application the size of the public exponent must be greater than $2^{16} = 65536$.

Remark : a small public exponent as 3 is proscribed.

Examples of asymmetric encryption

RSA cryptosystem (Rivest, Shamir, Adleman, 1978), based on factorization.

McEliece cryptosystem (1978), based on coding theory.

Merkle-Hellman cryptosystem (1978), knapsack problem based asymmetric cryptosystem, broken by Shamir.

ElGamal cryptosystem (1985), based on discrete logarithm.

Menezes-Vanstone cryptosystem (1993), based on elliptic curves.

RSAES-OAEP cryptosystem

Recommendation_ChiffAsym : the asymmetric encryption algorithms must dispose of a security proof.

Conformity : the asymmetric encryption **RSAES-OAEP** (RSA encryption scheme-OAEP), defined in PKCS#1 v2.1, is conform if Rule_FAC1, Rule_FAC2 and Rule_FAC3 are applied.

Process and semantic security of RSA-OAEP :

OAEP is used to process the plaintext before the asymmetric encryption (as RSA) and provides a probabilistic encryption :

$$m \rightarrow OAEP \rightarrow RSA \rightarrow c.$$

$$c \rightarrow RSA^{-1} \rightarrow OAEP^{-1} \rightarrow m.$$

OAEP

Optimal **A**symmetric **E**ncryption **P**adding (Bellare and Rogaway, 1994), recommended by PKCS#1, v2.1 (2002).

Process (computation) :

Let f_1 and f_2 two hash functions and r a random quantity.
Let m the plaintext with a classical padding (1000..0).

$$OAEP(m) = (m \oplus f_2(r)) \parallel (r \oplus f_1(m \oplus f_2(r))).$$

$\oplus$ means exclusive or and $\parallel$ means concatenation.

Process (recovery) :

Let $OAEP(m) = O_1 \parallel O_2$ with the size of O_1 is the size of m (that means $O_1 = m \oplus f_2(r)$ and O_2 the other part).

$$m = O_1 \oplus f_2(f_1(O_1) \oplus O_2).$$

Description of OAEP

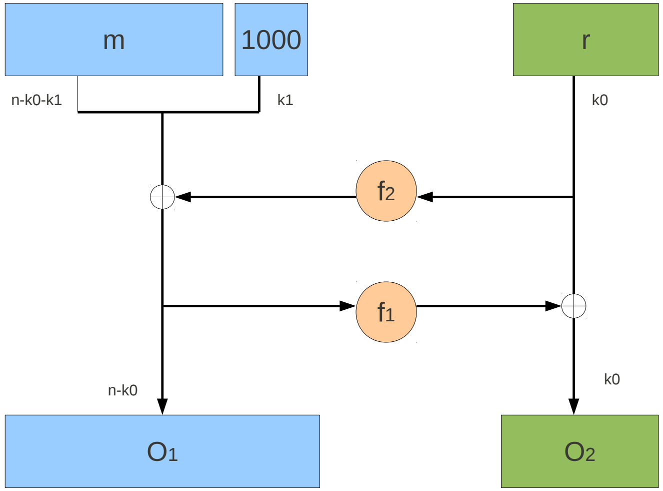

FIGURE: Description of OAEP

PKCS

PKCS means **P**ublic **K**ey **C**ryptography **S**tandards.

PKCS are developped and completely controlled by the company **RSA Security**, and consequently **are not standards.**

PKCS#1, ..., PKCS#15 are available at `www.rsa.com/rsalabs/` > standards initiatives.

Many standards are reformulated in RFC. For example :

RFC 3447 for PKCS#1.
RFC 2315 for PKCS#7.

Example of hybrid system : PGP

Pretty **G**ood **P**rivacy : encryption created by Zimmermann (1993), supporting authentication and integrity mechanism.

PGP follows the OpenPGP standard (RFC 4880).

Principle :

1. The message is encrypted with a symmetric encryption and a secret key K. Each key K is used one time and is called *session key*.
2. The key K is encrypted with an asymmetric encryption and the receiver's public key.
3. The encrypted message and the session key are send to the receiver, who decrypts the session key with his private key and the message with the session key.

Definitions on signature schemes

Digital signature :
A data associating a message to an entity (with a key).

Signature mechanism :
Algorithm realizing a digital signature (following a method processing data into a message to be signed).

Signature verification mechansim :
Algorithm verifying that a signature has indeed produced by a specific entity (with a method recovering data from a message).

Signature scheme :
A signature mechanism and a signature verification mechansim.

The **signer** who can realize easily electronical signature.

The **verifier** who can verify the signature without the knowledge of confidential informations.

Signature scheme with appendix

Signature scheme with appendix :

The signer uses his private key for the signature and sends the message with the signature and the public key to the verifier.

The verifier uses the public key to verify the signature (using the message).

Most of signature schemes are *with appendix*. For example : **RSA S**ignature **S**cheme with **A**ppendix.

Appendix vs message recovery :

With appendix means that the message is sent with the signature (to be distinguished from signature schemes with message recovery (Nyberg, Rueppel, 1993)).

Signatures schemes pre-process

The data which is signed is possibly very long.

Signature protocols use a **hash function**, which needs to be conform to the referential. The message m is hashed in $H(m)$.

Collision and second preimage attacks are directly related to signature security !

The hash value is **encoded** (or padded) before to be signed (in order to avoid existential attacks).

Examples of ad-hoc encoding systems are described in PKCS#1 v 1.5 or ISO 9796-2 (used by EMV).

Vulnerabilities on ISO 9796-2 encoding method (Coron, Naccache, Tibouchi, Weinmann, 2009).

Examples of asymmetric signature algorithms

RSA signature Scheme (Rivers, Shamir, Adleman, 1978), based on factorization.

El Gamal signature scheme (1985), based on discrete logarithm.

Schnorr signature scheme (1989), based on discrete logarithm.

DSA : **D**igital **S**ignature **A**lgorithm is a signature standard (called DSS, 1993) described in FIPS 186-2.

ECDSA (Vanstone, 1992), based on discrete logarithm on elliptic curves.

Asymmetric signatures conformity

Recommendation : asymmetric signature mechanism must dispose to a security proof.

Conformity : the signature mechanism **RSA-SSA-PSS**, defined in PKCS#1 v2.1 is conform to the referential if rule_FAC1, rule_FAC2 and rule_FAC3 are respected.

RSA-PSS is the combination of the RSA signature scheme with the PSS encoding method (Bellare, Rogaway, 1996).

Examples of provably secure signature schemes : **PSS** (**P**robabilistic **S**ignature **S**cheme) and **FDH** (**F**ull **D**omain **H**ash).

Conformity : the signature scheme **ECDSA** defined in FIPS 186-2 is conform to the referential if elliptic curves P-256, P-384, P-521, B-283, B-409 or B-571 are used.

PSS description (PKCS#1 v2.1)

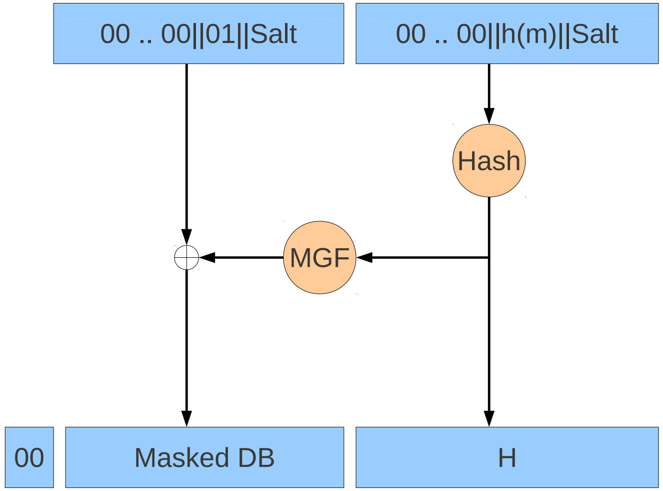

FIGURE: Description of PSS

ECDSA presentation

Elliptic **C**urve **D**igital **S**ignature **A**lgorithm.

Algorithm proposed by Vanstone (1992) and defined by the standard X9.62-1998.

Public keys : the elliptic curve E and a point P of the curve, of prime order n and the point $A = a.P$.

Private key : the quantity $a \in [1, n-1]$.

ECDSA algorithm

Signature algorithm of a message m :

1. Calculate $k \in [1, n-1]$ at random and $K = k.P$.
2. Let t be X_K (if $t = 0$ return to 1.).
3. Calculate $s = (m + at)k^{-1}$ mod n (if $s = 0$, return to 1.).

The signature of m is the pair (t, s).

Verification algorithm :

1. Check that t and s are in $[1, n-1]$.
2. Calculate $Q = m.s^{-1}P + ts^{-1}A$.
3. The signature is valid if X_Q mod $n = t$.

The random quantity k must be different for each signature of the same message.

Non-repudiation mechanism

Establishing the time when a digital signature was generated is often an important consideration (digital signatures timeliness).

Non-repudiation :

Digital signature timeliness are used for non-repudiation objectives and are described in NIST SP800-102.

Trusted Timestamp Authority :

Digital signature timeliness uses timestamps that are digitally signed by an authority : the Trusted Timestamp Authority (TTA).

The timestamp contains the (trusted) time and possibly other information.

Entity authentication

Authentication is used for the **verification of the identity of an entity** (preceded -or combined- by an identification process).

In computer security, **AAA protocols** means **A**uthentication-**A**uthorization-**A**ccounting.

Strong authentication corresponds to authentication, based on several factors, including :

1. Something that the user has (e.g. smart card, token).
2. Something that the user knows (e.g. password, PIN code).
3. Something that the user is (biometric).

Challenge/response protocols used for entity authentication are specified in FIPS 196 or NIST SP 800-78.

Access control matrix

Formal security model introduced by Lampson (1971).

The set of objects needed to be protected is denoted by O.

The set of subjects (users, process,..) is dentoted by S.

The set of rights denoted by R is of the form $r(s, o)$ where $s \in S$, $o \in O$ and $r(s, o) \in R$.

The *access control matrix* is the matrix $\{r(s, o)\}$.

A row of the matrix corresponds to an user and is called *capabilities*.

A column of the matrix corresponds to an object and is called *access control list*.

Authentication

The RGS considers three types of authentication :

1. Personal authentication with a login/password system in a local environment.
2. Personal authentication with a certificate.
3. Server authentication with a certificate.

Rule_Auth1 : authentication between two servers must be realized with a conform cryptographic protocol.

Rule_Auth2 : authentication between two servers must use a key management architecture (with electronical certificate).

Personal authentication with login and password

A login / password system is commonly used for authentication.

Password need to be carefully chosen and must be hashed (strengthen passwords with salt).

Information note of **CERTA** on passwords (**C**entre d'**E**xpertise gouvernemental de **R**éponse et de **T**raitement des **A**ttaques informatiques) :

`http://www.certa.ssi.gouv.fr/site/CERTA-2005-INF-001.pdf`

This type of authentication is a **weak authentication system** (replay attacks, entropy of password,..).

Entropy of password

Number of symbols	10 (ciphers)		26 (letters)		62 (ciphers and letters)		90 (all)		
Number of symbols by passwords	4	10	8	16	8	16	8	10	16
Key size (bits)	13	33	38	75	48	95	52	65	104

Challenge/response authentication

Authentication protocols usually use a **random challenge**, ensuring a protection agaisnt **replay attacks**, and are called *challenge/response protocols*.

Such authentication protocols use symmetric or asymmetric encryption or digital signature as cryptographic primitives.

Example with digital signatures :

1. Servers sends a random challenge c to the client.
2. Client signs the challenge c with his private key and sends it to the server.
3. Server controls the signature with the public key of the client (and the public key with a electronical certificate).

Mutual authentication

A mutual authentication protocol using a pair of public key/private key for each entities :

1. Alice sends a random challenge a to Bob.
2. Bob generates a random challenge b, concatenes the challenges in (b, a) and signs it with his private key. Bob sends the signature with b to Alice.
3. Alice verifies Bob's signature, concatenes (a, b) and signs it with his private key. Alice sends the signature to Bob.
4. Bob verifies the signature of Alice.

Exercice : compare this protocol with the STS protocol.

Zero-knowledge identification protocols

In a standard challenge/response protocol, the server obtains arbitrary signatures (or plaintext/ciphertext pairs).

Stronger identification protocols uses zero-knowledge proofs.

Examples of zero-knowledge identification protocols :

1. **Fiat-Shamir protocol** (1986).
2. **Guillou-Quisqater protocol** (1988).
3. **Schnorr protocol** (1989).

Plan

Reglementary references

Ordonnance n 2005-1516 du 8 décembre 2005 relative aux échanges électroniques entre les usagers et les autorités administratives et entre les autorités administratives

Décret n 2010-112 du 2 février 2010 pris pour l'application des articles 9, 10 et 12 de la précédente ordonnance.

Décret n 2001-272 du 30 mars 2001 pris pour l'application de l'article 1316-4 du code civil et relatif à la signature électronique.

Arrêté du 26 juillet 2004 relatif à la reconnaissance de la qualification des prestataires de services de certification électronique et à l'accréditation des organismes qui procèdent à leur évaluation.

Loi n 2000-321 du 12 avril 2000 relative aux droits des citoyens dans leurs relations avec les administrations.

ISO standards (1)

ISO 9564-3 :2003. Banking-Personal Identification Number Management and Security - Part 3 : Requirements for Offline PIN Handling in ATM and POS Systems, Nov. 2003.

ISO 16609 :2004. Banking - Requirements for Message Authentication Using Symmetric Techniques, Feb 2004.

ISO/TR 9564-4 :2004. Banking - Personal Identification Number (PIN) Management and Security - Part 4 : Guidelines for PIN Handling in Open Network, March 2004.

ISO/IEC 9798-1 :1997. IT- Security Techniques - Entity Authentication - Part 1 : General, July 1997.

ISO/IEC 9797-1 :1999. IT- Security Techniques - Message Authentication Codes (MACs) - Part 1 : Mechanisms Using a Block Cipher, December 1999.

ISO standards (2)

ISO/IEC 9796-2 :2002. IT- Security Techniques - Digital Signature Schemes Giving Message Recovery - Part 2 : Integer Factorization Based Mechanisms, Oct. 2002.

ISO/IEC 9797-2 :2002. IT - Security Techniques - Message Authentication Codes (MACs) - Part 2 : Mechanisms Using a Dedicated Hash-Function, June 2002.

ISO/IEC 13888-1 :2004. IT security techniques - Non-repudiation - Part 1 : General, June 2004.

ISO/IEC 18031 :2005. IT - Security Techniques - Random Bit Generation, November 2005.

ISO/IEC 19794-2 :2005. IT - Biometric Data Interchange Formats - Part 2 : Finger Minutiae Data, September 2005.

ISO standards (3)

ISO/IEC 14888-3 :2006. IT - Security Techniques - Digital Signatures with Appendix - Part 3 : Discrete Logarithm Based Mechanisms, Nov 2006.

ISO/IEC 7813 :2006. IT - Identification Cards - Financial Transaction Cards, June 2006.

ISO/IEC 9796-3 :2006. IT - Security Techniques - Digital Signature Schemes Giving Message Recovery - Part 3 : Discrete Logarithm Based Mechanisms, Sept. 2006.

ISO/IEC 14888-1 :2008. IT - Security Techniques - Digital Signatures with Appendix - Part 1 : General, April 2008

ISO/IEC 14888-2 :2008. IT - Security Techniques - Digital Signatures with Appendix - Part 2 : Integer Factorization Based Mechanisms, April 2008.