

RFC SYSTEM

RFC – Introduction

RFC is an Asymmetrical (public keys) cipher, which allows for the creation of safe keys to encrypt and decrypt a message.

RFC Has been designed primarily for:

- *Communications by the military;*
- *E-commerce;*
- *To completely withstand any form of mathematical attack;*
- *For long-term strategic security of data.*

RFC – The method (Basic principles)

Existing systems of public-key cryptography are based on factorization of large numbers or calculating the discrete logarithm (RSA, Diffie-Hellman, elliptic curves).

These systems of very large prime numbers are used to calculate the public and private keys, with a consequent increased demand in computing capabilities of the processor and in combination with specific algorithms for determining such numbers.

This translates into a greater complexity in the development of hardware and software for implementation as a means of factorisation and a greater demand in terms of resources of the device itself.

RFC Uses a Hash function in sum form.

The method is similar to that known as Diffie-Hellman/RSA but without the use of prime or large numbers in general.

This brings significant advantages from the point of view of computational speed and encryption.

The method by its very nature, is not subject to any known mathematical attack.

The only possibilities for compromise lay with the so-called Man-In-The-Middle (we can also avoid this threat) and the classic Brute Force attack.

Sum Hash Function in form RFC

The form of sum Hash uses an array $M()$ private values, a value of q private and form a p value of public form to create a digital public key (see Fig. 1)

The same values (the matrix $M()$ in the form $q \ e \ p$) are used to generate the encryption key.

The safety of RFC is based on the resolution of an array of random integers, which when passed to a particular function, produces an aggregate number (the public key) giving rise to an NP-complete problem to provide the cryptographic keys in the scheme:

$$y(ab) = F(h) \text{ mod } P$$

The two-dimensional matrix M () in the form q e p in RFC

RFC employs a two-dimensional matrix, the values of which start at a minimum of 2^8 , with the same applying to the choice of form q e value p. The number of rows in the matrix can be arbitrarily chosen.

*The value of p must be different from q e to ensure the creation of a congruence modular ($m(a) = m(A) * p$ and $m(b) = m(b) * p$) during the exchange of public keys for obtaining key municipality. (should this word be universality? - or, should it read "for obtaining a universal key arrangement?)*

Attack on RFC

Attempting an attack on RFC implies knowledge of the values of matrix M () and the module q e timeline generation values of the array. If the number of elements in the array is high enough (at least 32), the cost of an attack by brute force becomes prohibitive.

In simple terms, given the number (public key) -728191901186103 and from P knowledge, obtain the numerical sequence that expresses the encryption key. ^(c)

Authentication and Integrity with RFC

Also, can one define a grid user (one-off) from the corresponding known implemented legitimate transactions or secure messaging, while ensuring authentication and integrity with Phishing avoided as a secondary desired effect:

GRID FILTER			
5	1	7	9
10	4	2	3
8	3	11	15

For example, the grid shows that the numbers 7, 4, 3, 11 comprise the grid filter that allows you to use a subset of the keys produced (note that the number of keys that can be produced is substantial)
 This mode has a notable advantage: even if all keys in the grid are discovered, there remains the problem of solving the location of the sub-keys as defined in the diagram; i.e. those that would be used for phase cryptographic messages. It is also possible to use an 'derivation algorithm' to use for an infinite variation of some keys.

Particularities of RFC as a tool for authentication and integrity

Keys exchanged for the next phase of the cryptographic process are time constrained, this means that any keys which are exchanged and intercepted, have no real value and any examination of the real keys returns a null.
 The true key is composed by the legitimate communication, using the key time.

Moreover, the representation of the public key in "Format Space Defined by Random Points" is that it is difficult to trace the real numeric key and information contained therein (such as authentication and identification). In the figure below is an example representing the format of an obtained key.

Example by Key Points Defined by Random points (number Key is: -728191901186103)



This series of random points scattered in space without any logic, is a key time generated by specific mathematical processes.
 They enclose the information in a manner unintelligible: only legitimate owners and creators of the key are able to derive the information composing the real key.
 With this method you can share a key only once which is valid for ever!

RFC Sample

Procedures generation and key exchange:

Let's have the corresponding Alice and Bob example.

Alice generates her private keys and public keys: $ya = f(x)$

Bob generates his private and public keys: $yb = f(xb)$

Bob sends a message to Alice:

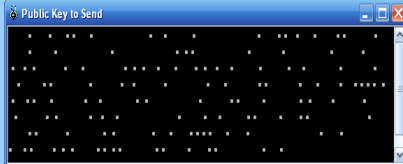
- To send a message to Alice, Bob uses a symmetric algorithm with the known key

Alice deciphers the encrypted message to Bob:

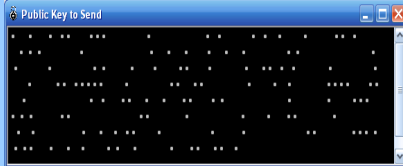
- Alice uses her private key to decipher the message which is accomplished by the reversal of the formula.

Temporary Public Keys representation:

PUBLIC KEY OF BOB (send to Alice)



PUBLIC KEY OF ALICE (Send to Bob)



Public and private values for Bob and Alice

User BOB		User ALICE	
m [[]] =	512 (# of private keys)	m [[]] =	512 (# of private keys)
N. SubKeys	512 (# of Encrypt keys)	N. SubKeys	512 (# of Encrypt keys)
P =	16180339 (Common Module)	P =	16180339 (Common Module)
ID =	8948E64D.9AE48139.386ED6AB.874	ID =	8948E64D.9AE48139.386ED6AB.874
GRID FILTER	11.221	GRID FILTER	11.221
Code...	Public Key Generation...	Code...	Public Key Generation...
q (module) =	1172418	q (module) =	4789597
x(0)=8154	y(0)=258091	x(0)=-102655	y(0)=-378230
x(1)=883237	y(1)=572414	x(1)=-527742	y(1)=461605
x(2)=881700	y(2)=-41505	x(2)=50616	y(2)=920499
x(3)=783116	y(3)=517064	x(3)=502874	y(3)=889042
x(4)=701630	y(4)=-928296	x(4)=-81816	y(4)=-827954
f(x1)=	959355950354	f(x1)=	-438379645348
f(x2)= K+	f(x2)= K+
Temporary Public Key	Temporary Public Key
Test Encryption	0	Test Encryption	0
2009/03/03 11:22:23		2009/03/03 11:22:21	
Error Simulation	no Error	Error Simulation	no Error
Keys of: BOB	Private Rand Token	Keys of: ALICE	Private Rand Token
1: 12601035	T1: 1406402665	1: 12601035	T1: -243620797
2: 869646	T2: -1835262808	2: 869646	T2: 1557363977
3: 5601373	T3: -243620797	3: 5601373	T3: 1406402665
4: 4492822	T4: 1557363977	4: 4492822	T4: -1835262808
5: 3913309	-----	5: 3913309	-----
6: 11556422	x: 884883037	6: 11556422	x: 884883037
7: 3624294		7: 3624294	
8: 15472334		8: 15472334	
9: 15490116		9: 15490116	
10: 6682721		10: 6682721	
11: 5215185		11: 5215185	

As we see from the illustration above, the key to encrypt/decrypt the sequence is represented in the figure by k (0) at (11)

(c) Note that unlike the example, the key can be constructed of millions of umbers!

Similar to quantum cryptography, each key number representing one or more photons in a state, x.

COMPARATIVE TABLE OF PUBLIC KEY SECURITY SYSTEMS

+/-	QKD	+/-	Public Key – RSA/DH/Elliptic Curves	+/-	Public Key RFC
-	Requires hardware and dedicated communication lines	+	It can be implemented in Software/Hardware and portable applications	+	It can be implemented in Software/Hardware and portable devices
+	Absolutely safe because based on fundamental laws of quantum physics	-	Integrity not decided, based on mathematical known problems for which there is not a publicly known simple solution (which may, of course, exist)	+	mathematically based on a numerical aggregation (sum nonlinear) problem for which there is no algorithm available to obtain the inverse value.
+	Security is based on general principles and does not require future amendments	-	Security is based on key increasingly depending on the scope and power of computers	+	Security is based on general principles and does not require future changes. The philosophy and principles (in a mathematical sense) equal quantum cryptography.
+	QKD is secure even in the presence of quantum computers	-	Quantum computers will be able to find the keys and then break the system (see Quantum Computers for decrypting codes)	+	Very complex even for a quantum computer to break this system.
-	Limited availability and very expensive	+	Average to low cost which depends upon the manner of implementation and use	+	A provision of mass and low cost regardless of deployment and use
-	Brand new system and great development for the future.	+	Widespread implementation and experience with the solution.	-	Little experience thus far, but by its mathematical nature, can be seen as a guarantee of exceptional security
-	To date only works over limited distances (max 200 to 300Km) and requires direct connections with fibre optic cabling	+	It works at any distance and with any type of network	+	It works at any distance and with any type of network
-	The speed of creating key is still low (but it is growing very fast)	-	Substantial computational resources needed to ensure adequate security and is therefore liable to simple attack (very dangerous)	+	Requires low computational resources to ensure high security standards and is not subject to any known type of attack.
+	Can be easily used with a OTP (One Time Pad) algorithm that guarantees excellent security.	-	Cannot be easily used with OTP algorithms.	+	Can be easily used with OTP (One Time Pad) algorithms (indeed, specifically optimised for use with this algorithm) the only one that can guarantee today an excellent security.

+ Point in favour - Against