CRYPTOSYSTEM ARCHITECTURES FOR VERY HIGH THROUGHPUT MULTIMEDIA ENCRYPTION: THE RPK SOLUTION

A. Romeo, G. Romolotti, M. Mattavelli, D. Mlynek

Integrated System Center (C3I)
École Polytechnique Fédérale de Lausanne CH-1015 Ecublens, Lausanne, Switzerland

ABSTRACT

This paper presents a new approach to secure data encryption for high bit-rates communications. The proposed scheme is based on RPK, a new cryptographic algorithm based on the discrete logarithm problem implemented by a discrete exponentiation over finite Galois fields GF[2^n] and the Alternating Stop and Go as a Stream Cipher Block. The paper describes RPK architecture solutions yielding to low-complexity implementations, particularly suitable for encrypting high bit-rates multimedia contents. A comparison with alternative encryption schemes such as the classical RSA shows the implementation advantages of the RPK approach.

1- INTRODUCTION

Secure transmission requirements are not an original issue of the contemporary world, indeed solutions to these problems have been pursued for centuries. Some examples from ancient or recent history, such as the messages used by Julius Cesar and the Enigma code during the Second World War, are signs of this interest that in the past was usually limited to applications in the military field.

Nowadays, the increasing use of transmission of digital content over communication networks, of which the success of the world-wide-web is the clearest example, raises new security issues in transmission. This fact is the main reason of the growing research interest in the evolution of cryptographic algorithm with application to secure digital transmissions. For instance, the birth of electronic trade has been giving a great development to cryptography. An important security issue of the electronics trade is the non-repudiation mechanism. The goal is to ascertain the operator identity and to establish a mechanism in order to ascribe responsibility of action to the operator. The solution of this problem is the main reason of the recent introduction of public-key crypto systems. Their main characteristic is the one-way functionality concept. A one-way-function is a function f(x) for which it is easy, known x, to compute f(x), but it is significantly much harder (and ideally not possible in a reasonable amount of time) to perform the inverse operation. Unfortunately *state of the art* public-key crypto systems with high security transmission performance require high processing resources when applied to high bit-rates and result not suitable for modern multimedia communications.

The paper is organized as follows, the second part of this section briefly summarize the main features of public-key and symmetric key systems. Section 2 describes in more details the RPK algorithm, with a particular attention to the innovative features that are particularly interesting for high throughput applications, while the last section reports a comparison between RPK and RSA implementation features. Public-key crypto systems can be divided in 4 main groups defined by the problem, needed to be solved, to crack the system:

PROBLEM	ALGORITHM
Factorization problem	RSA
Discrete logarithm	Diffie-Hellman
Elliptic Curves	Elliptic Curves
Knapsack algorithm	Knapsack algorithm

The most known and used algorithms belong to the first, second, and third group, while the fourth group of algorithms is not very popular since it has been successfully attacked in different ways. By employing an adequate key bit length, the first three groups of algorithms can be considered very secure. All of them operate on fixed length data blocks, where the public/private key is used to crypt/decrypt the message. We refer to these classes of systems as “block systems”. Unfortunately, the main drawback for the applications of these crypto systems at high bit-rates is the complexity of the encryption/decryption processing which results directly proportional to the number of bit to be processed.

Another class of crypto system is known as “stream cipher systems”. The stream ciphers systems operate with a time-varying transformation on individual plain-text digits. A stream cipher uses a short key to generate the key-stream, which appears to be random. Using, for instance, a non-linear combination of Linear Feedback Shift Register (LFSR) it is possible to make a good pseudo-random key-stream generator. Stream ciphers systems can operate very fast on the bit-streams are relatively easy to implement; and generally result much faster than block ciphers. Conversely stream ciphers are symmetric-key systems and, for this reason, they are unable to provide secure non-repudiation mechanisms [1].

The most popular crypto systems are based on hybrid method, i.e. an asymmetric algorithm to initialize a symmetric one. This is a compromise between key security key, management and the speed of the stream cipher systems or hash functions generation.

2- THE RPK ALGORITHM

Crypto systems based on blocks such as RSA or Diffie-Hellman, as discussed in the previous paragraph present a high level of security, but require large computational resources when applied to high bit-rate tasks. Conversely, stream ciphers systems are low cost and fast, but they cannot be considered secure enough against attacks.

The RPK approach to public-key encryption combines the best characteristics of both classical crypto block systems and stream ciphers systems yielding a secure system that requires very low processing resources at very high throughput rates.

All RPK system basic operations are mathematically equivalent to the exponentiation in finite fields [4]. The private and public operation scheme of RPK is depicted in Figure 1.

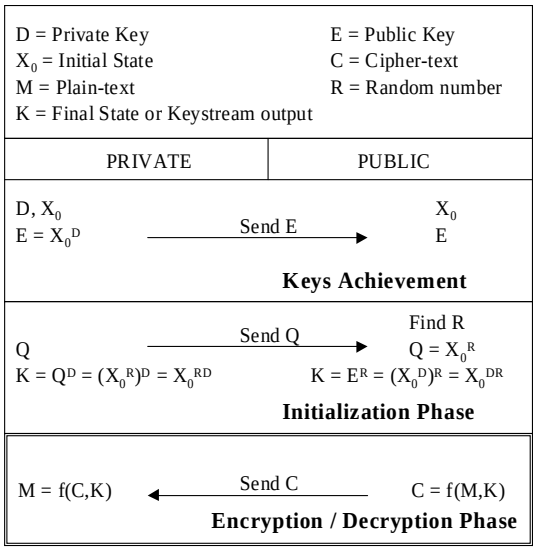

Figure 1. *Schematic representation of the RPK public key Crypto-system.*

Main features of the scheme are:

- The RPK system is based on the discrete logarithm problem. This means that in this system a public key is calculated from a private key using operations mathematically equivalent to exponentiation in finite fields.
- For reasons of computational efficiency, speed and security, the finite fields underlying the RPK system are the Galois fields GF[2^p] where p is a Marsenne prime number (i.e. p and 2^p-1 must be both prime numbers).
- The choice of the private key is made without any restriction, thus making possible the creation of the private key with a simple random generator.
- The cipher-text has exactly the same length of the plain-text, provided that a short header block enabling the correct set-up of the system for the decryption is transmitted before the cipher-text.
- Each state of the machine has no memory of its history. In other words, if we place the machine into any particular state and then it evolves to successive states by means of “clock” cycles, the sequence of outputs will be the same regardless of how the machine happened to arrive at that state.
- RPK is a non-deterministic crypto-system therefore if the same key is used to encrypt a given plain-text the resulting cipher-texts will differ in a non-systematic way, ideally in a random fashion.
- Crypto-system can also be classified according to whether they are block system or stream system. In the case of RPK the classification cannot be applied, since it possesses some of the characteristics of both systems. However, it is probably more appropriate to consider the RPK system as a stream system.
- Encryption and decryption do not commute in this system; however it is possible to use alternative methods for secure signatures and/or authentication.

3- THE RPK SYSTEM

The RPK system involves two phases, an initialization phase and a combining or streaming phase. These are not two separate cryptographic algorithms, but instead are best viewed as two connected parts of the RPK approach, unified by the concept of a Mixture Generator. The Mixture Generator is a pseudorandom binary key-stream generator. The probability distribution of the mixture, in the case when there are finite number of random variables being mixed, is a "convex combination" of the distributions of the components. It is easy to show that a mixture of independent identically uniform-distributed random variables is itself uniform [4]. The mixer outputs, or its states, are used to select, in a memory-less fashion, outputs from members of a set of other component pseudo-random binary generators. The Mixture Generator and the other components are maximal-period linear shift register generators (MLSRGs). When using MLSRGs as component generators, it is essential to use generators with the mathematical property that their generator polynomials are primitive polynomials. An important property is that they may have a prime number of stages (2^n-1), so that the bit lengths of the keys must be Marsenne primes.

Usually, the RPK encryption/decryption process [5] begins with the transmission of the Public Key, which can be known by everybody. The first and the second diagram of figure 2 depict “the Public” steps of the process during the beginning of encryption.

“The Public” operator first generate a “true” random number R. The diagram illustrates using the finite state machine representation how to reach the state Q that is R steps away from the initial state X_0. “The Public” then sends Q to “the Private” key owner as shown in the scheme of figure 1.

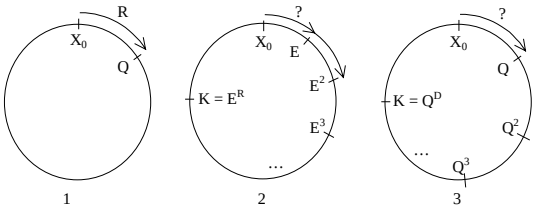

Figure 2. *The Mixture Generator.*

"The Public" reset the machine and then loads it with the state E represented by the public key. Despite the fact that the number of steps needed to get to E from the initial state is unknown (private key) "the Public" is able to use the machine to jump to a state R times further along in the state sequence than E is. "The Public's" machine has now arrived at a state K, mathematically represented by E^R, that is RD (R times D) steps away from the initial state. This completes the RPK initialization phase and "the Public" is ready for the stream cipher system.

The third diagram depicts "the Private" steps representing the initialization process.

"The Private" first receives Q and sets the machine with the state represented by Q. Then, using the same special abilities of the machine, the recipient jumps to a state D times further along the state sequence than Q. Here "the Private" knows Q and D, but does not know R (the random number selected by "the Public" specifically for this message). Nevertheless, the recipient's Mixture Generator has now arrived at a state mathematically represented by Q^D that is also RD steps away from the initial state. That is, it has managed to arrive at the same state K. Having completed the initialization phase, "the Private" is now ready to initialize the stream cipher system to find the key-stream's output sequence as "the Public" and to decrypt the cipher-text received.

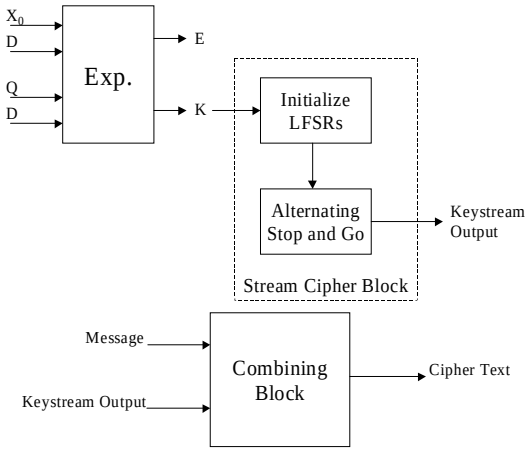

Figure 3. *Scheme of the architecture for the implementation of high bitrate RPK crypto systems.*

The advantages of mixer generator configurations are that their periods are very long, their distributions of zeros and ones is well-balanced, and successive outputs are substantially non-correlated. Theirs outputs also have excellent statistical properties in terms of their n-tuple distributions and runs statistics.

An example of mixture-generator is the alternative stop-and-go [3] as shown in Figure 4. This architecture combined with the RPK private/public key scheme provides an extremely secure and low-complexity implementation of a complete crypto-system.

Considering the extremely long periodicity of such generators (i.e. of the orders of 2 power 607) it is not feasible to crack the system by actually run them long enough to extract the content of each MLSRGs. Usually, even very high bitrate multimedia transmissions explore only a tiny fraction of the number of states required to decrypt and possibly crack the system.

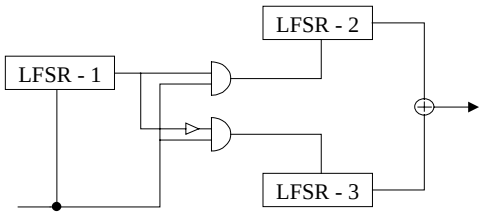

Figure 4. *Scheme of an alternating stop-and-go mixture generator.*

A highly compact and efficient method for calculating the future state of single MLSRG exists. It is based on interpreting the contents of the stages of the register as coefficients of a polynomial in one "indeterminate" x. Using this interpretation over the Galois fields [2] it is possible to reduce the exponentiation operation to a much easier polynomials multiplication. This operation, in practice, is very simple and can be efficiently implemented using the shift register itself.

The other steps to encrypt the message are based the same type of operations, the only difference is that the exponent become the initial state and vice versa. Finding the final states K, after the initialization phase, is possible to use the mixture generator and generate the key-stream.

4- COMPARISON RPK AND RSA+3DES IMPLEMENTATION COMPLEXITIES

Figure 5 shows typical qualitative curves of the processing time/complexity of RPK and RSA+3DES versus the bit/bit-rate of the plain/cipher-text at the same security level (RPK key length = 607 bit; RSA key length = 512 bit). We can notice that in the case of RPK after an initial "setup time" this machine can encrypt and decrypt messages very efficiently. In fact once the machine has been initialized, the only factor that determines the bit rate output capabilities of a RPK system is the complexity of the stream-cipher and plain/cipher-text mixing.

Assuming to use the "alternative stop-and-go" as mixing algorithm [3] and architecture the output bit rate can be the same order of magnitude than the clock of the internal machine. The only operations necessary for this type of mixing are the passage to a future state in one of the two LFSRs (only a shift register operation); and the

xor operation between one bit of the key-stream and the each bit of the plain/cipher-text in the Combining Block.

An extremely high level of security can be guaranteed since long private keys such as for instance with a 1279 (Marsenne prime) bit length do not affect the throughput performance, but only increase the set up time. Figure 5 illustrates the fact that the RPK curves slope only depends on the complexity of f(M,C). The figure also shows that the RPK system is less appropriate in terms of complexity for short messages. In fact, the start-up time is larger than the RSA+3DES encryption/decryption time, but with the increase of the message length the RPK system becomes rapidly convenient.

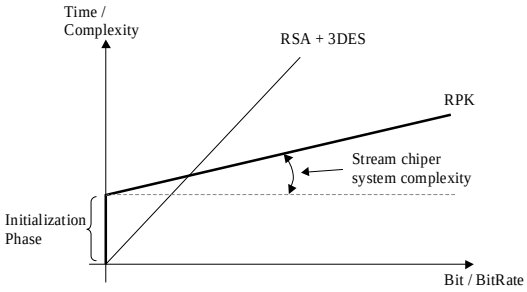

Figure 5. *RSA+3DES - RPK performance comparison*

The threshold only depends on the first RPK system phase, the initialization.

4.1- Software Implementation

It is possible to reduce the start up time introducing a memory to the basic implementation of RPK systems. The representation of the polynoms necessary to compute the exponentiation can be pre-computed and stored in memory yielding processing reductions of the initialization phase of about 70% for 607 key lengths (45Kbytes) and even higher for longer keys. A complexity analysis shows that the largest part of the processing is given by the polynoms multiplication and the "shifter operation". The diagram in Figure 6 reports the number of operations for each data type of the complete algorithm (left column) while the right column reports the fraction corresponding to of the "shifter" function.

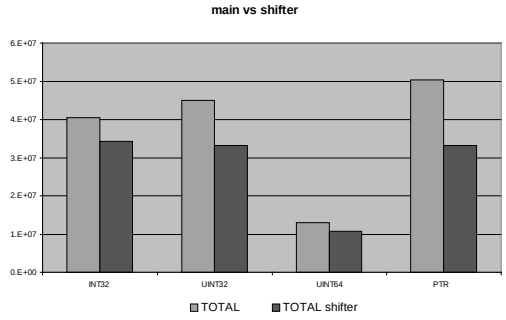

Figure 6. *Number of operations for a 607 key-length RPK initialization phase based on processing data of 32 and 64 bits. Left columns, total number of operations; right column number of "shift" operations.*

4.2- Hardware Implementation

The operation to be executed in the initialization phase of RPK is the exponentiation in $GF[2^n]$. The exponentiation can be implemented in hardware by means of a "square module" and a "multiplication module". To execute an exponentiation A^D, assuming $A = a_0 2^0 + a_1 2^1 + \ldots + a_n 2^n$ and $D = d_0 2^0 + d_1 2^1 + \ldots + d_n 2^n$ the exponentiation results: $A^D = A^{d_0 2^0} \cdot A^{d_1 2^1} \ldots \cdot A^{d_n 2^n}$, it means executing squares (A, A^2, A^4, A^8,…, $A^{2\wedge n}$) and multiplication between the last result and the next square only if the correspondent D-bit is one.

The square (mod p(x)) can be implemented in a single clock cycle as depict in the figure7.

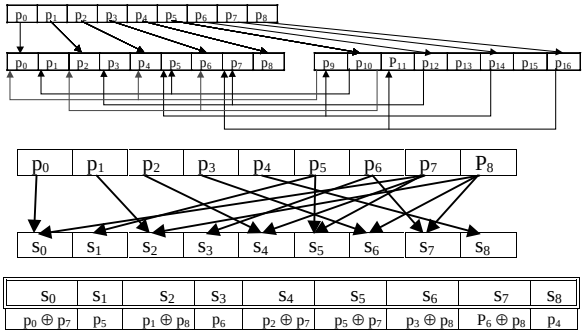

Figure 7. *Example of square with 9 bits and* $p(x)=1+x^4+x^9$.

5- CONCLUSION

This paper presents a new public-key encryption approach for high bit-rate applications suitable for secure multimedia communications. RPK implementations are based on an initialization phase and a low complexity streaming phase that is the only stage of the algorithm concerned by high bit-rate processing. These features make RPK system an advantageous solution to secure multimedia digital content transmission when compared to the classical RSA+3DES systems.

This work was done in collaboration with *RPK Securité SA, Suisse.*

REFERENCES

[1] Bruce Schneier "*Applied Cryptography*". John Wiley and Sons, New York, 2nd edition, 1996.

[2] Rudolf Lidl, Harald Niederreiter "*Introduction to finite fields and their applications*", Addison-Wesley Publishing Company, 1983

[3] C.G. Gunther, 1988, "*Alternative Step generators controlled by de Bruijn sequences*" Advances in cryptology - EUROCRYPT '87, p. 5-14

[4] "*The RPK public-key cryptographic system Technical Summary*" and "Detailed Supplemental Technical description of the RPK public-key cryptographic system" http://www.rpkusa.com

[5] U.S. patent number 5,799,088, August 25, 1998 or New Zealand patent number 277,128 on August 17, 1998.

[6] Menezes, Oorschot, Vanstone "*Handbook of Applied Cryptography*". CRC 1997.