Comments on the security of a hybrid quantum-classical authentication protocol.

Manoel S. S. Azevedo, Rex A. C. Medeiros, Raimundo C. S. Freire and Francisco M. de Assis
Universidade Federal de Campina Grande, Departamento de Engenharia Elétrica Av. Aprígio Veloso, 882, Bodocongô, 58109-970 Campina Grande-PB, Brasil [mazevedo,rex,freire,fmarcos@dee.ufcg.edu.br]

Abstract—In 1982 Brassard [24] introduced message authentication code (MAC) based on only one hash function and a pseudo-random sequence generator. For each message first an authentication tag is created with application of the hash function and after the tag created is XORed with the pseudo-random sequence. The pseudo-random generator has its security based on the hardness of number theoretic problems like factoring Blum numbers or solving discrete logarithms. In his original paper Brassard suggested the use of the Blum-Micali pseudo-random sequence generator. Clearly the security of the authentication algorithm depends on the hardness of the mentioned problems. However, the advent of quantum computation offers a serious menace to this because it is shown that such class of problem can be polynomially solved by quantum computers [23]. In [1] Medeiros et al. shown that if the authentication tag was quantumly encoded with help of a Blum-Micali pseudo-random sequence generator, the security of the Brassard scheme is hold. In this article we remark by a simple argument that the security of the authentication algorithm depends on the hardness of the mentioned problems. However, the advent of quantum computation offers a serious menaces to this because it is shown that such class of problem can be polynomially solved by quantum computers [23].

In 1982 Brassard [24] has proposed a technique for message authentication that use a “short key” $K = (x_0, h)$ where $x_0$ is a seed for a pseudo-random sequence generator (PRSG) and $h$ stands for an universal hash function. The main idea of the Brassard protocol is that of creating a pseudo one-time authentication tag, $a(m, n) = h(m) \oplus x_0(n)$, where $m$ is the n-th message transmitted and $x_0(n)$ is a block of bits of the pseudo-random sequence with the same length of $h(m)$. The security of the technique is supported by the security of the PRSG. In his original article, Brassard has suggested employ the Blum-Micali (BM) pseudo-random bit generator. The majority of the PRSGs have their security based on number theoretic problems that are recognized as computationally hard. This the case of factorization of Blum numbers (defined below) and the case of discrete logarithm calculation. In a classical environment, that is, with no quantum computer availability, the supposition can be accepted [3]. However, if the attacker owns quantum computation power assumption is no longer valid because of that mentioned number theoretic problems are demonstrated to be solvable in polynomial time by a quantum computer [23], [19].

Manoel S.S. Azevedo, Francisco M. de Assis e Raimundo C. S. Freire are with UFCG, Campina Grande, Brasil, Rex A.C. Medeiros is with ENST, Paris

In other to circumvent limitation of the Brassard authentication scheme in case of the eavesdropper with quantum computation power, Medeiros [1] have proposed create another pseudo-random sequence (with seed $y_0$) of a two-dimensional Hilbert space to quantumly encode the classical authentication tag $a(m, n)$ (we call MA4 this new procedure). The scheme it is displayed schematically in Figure 1. Intuitively because the authentication tag is transmitted encoded in non-orthogonal quantum bases the attacker is impaired to get enough bits to determine the seed $y_0$ of the second PRBG.

In this paper the MA4 is modified with replacement of the BM PRBG by the BBS PRBG. The key advantage of the BBS over the BM is that of generate more bits per interaction [21]. The main contribution of this paper is to show that security is hold with the replacement of the PN bit generators.

Throughout we use the following notation:

- $|\psi\rangle, \langle \psi |$ vectors (ket, respec. bra)
- $|v\rangle, \langle \omega |$ Inner product between the vectors $|v\rangle$ and $|\omega\rangle$.
- $|v\rangle \otimes |\omega\rangle$ outer product between the vectors $|v\rangle$ and $|\omega\rangle$.
- $\rho$ density operator.
- $\delta_{ij}$, function that assume value 1, if and so if, $i=j$, otherwise assume value 0.
- $A^\dagger$, Hermitian conjugate or adjoint of the matrix $A$.
- $I$, identity matrix.
- $|\psi\rangle^{\otimes n}$, n-fold tensor products of the state $|\psi\rangle$.
- POVM, Positive Operator-Valued Measure.
- PRS(B)G, Pseudo-random sequence (bit) generator.
- $s \in S$ indicates that $s$ is chosen uniformly at random from set $S$.
- $N$ is a Blum integer, i.e., $N = pq$, where $p, q$ are prime, $p \equiv q \equiv 3 \mod 4$.
- $Q_{R_p}$, set of the quadratic residues mod $p$.
- $Q_{N_{R_p}}$, set of the quadratic not-residues mod $p$.
- $Z^*_n(+1)$, set of the integers with Jacobi symbol +1 mod $N$.
- $\Lambda_N = \{0, \ldots, N\}$
- $[0, N/2) = \{1, 2, \ldots, [N/2]\}$
- $\tilde{U}(y)$ denotes the i-th least significant bit of $y$, $i = 1, 2, \ldots$
- $E_N(y) = [y^2 \mod N]$, is the absolute Rabin function [21, pp.356]

The paper is organized as follows. The equivalence in terms of security between the BM and BBS pseudo-random sequence generators is reviewed in the next section. In section III we describe the authentication protocol with replacement of the BM PRBG by the BBS PRG. In section IV the analysis of security of the protocol is presented, finally section V concludes the paper.
II. COMPARISON BETWEEN BLUM-BLUM-SHUB AND BLUM-MICALI SEQUENCE GENERATORS

An important setback related some random number generators is that of its inefficiency in the sense only one secure pseudo-random bit is obtained per modular multiplication. If \( n \) is the length of the seed (in bits) implies that \( n^2 \) algorithmic steps are required. In practice pseudo-random number generators output \( n \) bits per multiplication, in this context Vazirani et all [18] has shown that a quadratic residue based algorithm, the Blum-Blum-Shub pseudo-random bit generator (BBS PRBG) allows generate \( O(\log \log n) \) secure pseudo-random bits per modular multiplication. They also shown that “invert” such that algorithm (find out seed) is so hard as factoring Blum numbers. For completeness we present below the pseudo-code of the BBS PRBG algorithm [21].

**Algorithm - BBS pseudo-random generator**

**Input:** \( x_0 \in \mathbb{Z}_N \), \( k \), \( j \) positive integers.

**Output:** A binary sequence \( \{ b_1, b_2, b_3, ..., b_M \} \) of length \( M = jk \).

**Initialization:** \( x_1 := x_0 \).

**Iteration:** for \( i = 1, ..., k \) do for \( r = 1, ..., j \) do

\[
\begin{align*}
    b_{(i-1)j+r} &= l_{j-r+1}(x_i) \\
    x_{i+1} &= E_N(x_i)
\end{align*}
\]

We remark that the algorithm above generates \( j \) output PN bits per iteration, iterating the absolute Rabin function \( k \) times generating \( j \) bits on each iteration.

**Example:** Given, \( N = 133 \), \( r = 6 \), \( j = 6 \), we will have

\[
\begin{align*}
    i &= 1 \\
    x_1 &= \{b_1, b_2, b_3, ..., b_M \}
\end{align*}
\]

Thus, continuing until the \( i = 6 \) iteration, the following sequence is generated:

\[
b_1 \ldots b_{36} = 00100110011101110001000000010110111.
\]

In [5] Sidorenko et al has shown that a successful attack on BBS generator is equivalent to a successful attack on the problem of factoring a Blum number. Therefore in a classical context employ Blum-Micali generator or BBS are equivalent under the point of view of computational security. In the same paper it is considered a more realistic set up where running time is included in the definition of the security of an algorithm. For sake of clearness we repeat the following well-known notion of statistical test for pseudo-random sequences. Consider a probabilistic algorithm \( A \) (statistical test) such that a binary sequence \( b = b_1, ..., b_M \in S \subseteq \{0, 1\}^M \) outputs a bit \( A(b) \in \{0, 1\} \).

**Definition 1:** A PRBG passes statistical test \( A \) with tolerance \( \epsilon > 0 \) if

\[
\left| \Pr \left( A(b) = 1 \mid b \in \mathbb{R} S \right) - \Pr \left( A(b) = 1 \mid b \in \{0, 1\}^M \right) \right| < \epsilon
\]

Otherwise the PRBG fails statistical test \( A \) with tolerance \( \epsilon \). The probability is taken over all choices of \( b \), and inner random coin flips of \( A \).

We remark that \( b \in \{0, 1\}^M \) stands for a truly random binary \( M \)-sequence.

**Definition 2:** A pseudo-random generator is \( (T_A, \epsilon) \)-secure if it passes all statistical tests with running time \( T_A \) with tolerance \( \epsilon \).

The Blum-Micali generator [20](BM) is based on the computational intractability of the discrete logarithms problem. Such that one-way property means that it is hard to find the complete preimage \( x \) of a value \( f(x) \), to the secrecy of individual bits of \( x \), i.e., that an attacker cannot get partial information. The basic notion here is that of a hard-core bit [1].

\[
y = g^x \mod p
\]

where \( p \) is an odd prime and \( g \) is a generator of \( GF(p) \) and \( x \in GF(p) \). Given \( p, g, \) and \( x \), calculation of \( y \) from \( x \) is direct. However, given \( y, g \) and \( p \) to find \( x \), he is one problem of the discrete logarithms mod \( p \).

The generating BBS [21] also it is based on the computational intractability, given for the equation

\[
y = x^2 \mod N
\]

where \( N = p \cdot q \) and \( p \) and \( q \) are both the 3 congruent modulo 4, this are reduced to classic problem of the search of an efficient algorithm of factoring that is considered computationally hard. It is assured by the corollary below, found in [21].

**Corollary 1:** Suppose the BBS pseudo-random generator is vulnerable against a \( (T_A, \epsilon) \)-state recovery attack \( A \). Then there exists a probabilistic algorithm \( F \) that factors the modulus \( N \) in expected time

\[
T_F = 2e^{-1}T_A + O(\epsilon^{-1}n^2M)
\]

We remark that the corollary above establishes an equivalence between the attack to the BBS pseudo-random generator to that of factoring an integer.

**A. Order-finding and hidden subgroup problems**

Let \( x \) and \( N \) be positive integer with no common factors and such that \( x < N \). The order of \( x \) modulo \( N \) is defined to be the least positive integer \( r \), such that \( x^r \equiv 1 \mod N \). The order-finding problem is to determine the order for some specified \( x \) and \( N \). This problem is considered intractable on
a classical computer. A quantum algorithm to solve the order-finding problem using polynomial resources was proposed by Shor [25]. This algorithm makes use of a procedure known as phase estimation, which consists of estimating the phase \( \phi \) of an eigenvalue \( e^{2\pi i \phi} \) associated with a particular eigenvector \( |\psi\rangle \) of a unitary operator \( U \).

Shor also demonstrates that the hidden subgroup problem (HSP) for a finite Abelian group \( G \) can be solved by a quantum computer using a number of operations polynomial in \( \log |G| \). Moreover, he showed that the HSP can be reduced to the order-finding problem. Factorization, discrete logarithms, period-finding, and many others problems are instances of the hidden subgroup problem.

The main aspect we point here is that for any of these problems it is necessary to have explicitly the classical parameters in order to solve them using a quantum computer. For example, to solve the discrete logarithm problem, we must know the integers \( a \) and \( b \equiv a^x \) in order to construct the unitary operation used in the order-finding algorithm [23], [9].

We conclude from discussion above that the BM and BBS PRBG equivale in terms of hardness and therefore they equivale in terms of “classical” security. Inspite of such that equivalence, the BBS algorithm is more efficient than the BM algorithm so is capable to produce \( \log \log (m) \) bits for one modular square, while the BM generator produces just one for each modular exponentiation [5].

### III. Blum-Blum-Shub Hybrid Authentication Protocol

The algorithm studied here is the same proposed by Medeiros [1] with only one modification that is the replacement of the Blum-Micali PRBG by the Blum-Blum-Shub PRBG, the reader is referred to that paper for details. The protocol uses another private key here \( y_a \), that it is a seed for the same generating of pseudo-random sequences. When Alice desires to send a message certified to Bob, it carries through all the described steps for protocol of MA4. Considering that Alice and Bob share a key private that it consists of a function hash particular \( h \in H \) and two seeds \( x_a \) and \( y_a \). For n-th message \( m \in M \) to be sent, Alice prepares a tag \( a(m, n) \) of \( k \) bits, given the equation

\[
a_B(m, n) = h(m_B) \oplus x_a(n),
\]

where \( x_a(n) = x_a[(n - 1)k + 1, \ldots, nk] \). Alice creates \( k \) qubits in bases \( Z \) or \( X \), depending on the tag \( a(m, n) \) and of the sequence \( y_a(n) \). It sends qubits through a perfect quantum channel. The message is sent by a channel unsafe, being able to be classic or quantum. After that Bob chooses the bases (joint POVMs) used in the measurement in accordance with the pseudo-random sequence \( y_a(n) \). As result, it gets a sequence of \( k \) bits, \( a_B(m, n) \). Now Bob calculates a local tag \( a'_B(m, n) \), based in the message received and in sequence \( x_a(n) \), that it is compared with \( a_B(m, n) \). If the tags are identical, Bob considers that the message is authentic, in case that contrary, it rejects.

The security of the protocol is assured by the generator of sequences pseudo-random (PRGB), the BBS, that already was described previously.

The detailed description more of the protocol is seen as it follows: it considers that Alice e Bob combines to use two orthonormal bases for the space of Hilbert of dimension two, sharing of a private key that consists of a function hash particular \( h \in H \) and two seeds \( x_a \) and \( y_a \), the protocol for authentication quantum of classic messages it can be summarized as it follows:

For n-th message \( m \in M \), Alice generates a label given for \( a(m, n) \)

\[
Z = \{ |0\rangle, |1\rangle \}
\]

\[
X = \{ |\pm\rangle, |\mp\rangle \},
\]

where

\[
|+\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}} \quad \text{e} \quad |\mp\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}}.
\]

These are the same used orthonormal bases employed to create the four states quantum in protocol BB84 [3]. For each bit of \( a(m, n) \), Alice it prepares a quantum state not entangled \( |\theta_{n_j}\rangle \), that it is based on the bit correspondent emitted for the generating BBS with seed \( y_a \). Of this form, if \( j \)-th bit of \( y_a(n) \) is zero, Alice prepares \( |\theta_{n_j}\rangle \) using base \( Z \) of following way:

\[
|\theta_{n_j}\rangle = \begin{cases} 
|0\rangle & \text{if the } j\text{-th bit of } a(m, n) \text{ is } 0 \\
|1\rangle & \text{if the } j\text{-th bit of } a(m, n) \text{ is } 1
\end{cases}
\]

Similarly if \( j \)-ésimo bit of \( y_a(n) \) is 1, Alice prepares \( |\theta_{n_j}\rangle \) using base \( X \), where

\[
|\theta_{n_j}\rangle = \begin{cases} 
|+\rangle & \text{if the } j\text{-th bit of } a(m, n) \text{ is } 0 \\
|-\rangle & \text{if the } j\text{-th bit of } a(m, n) \text{ is } 1
\end{cases}
\]

After the qubits generation, Alice sends the state \( |\theta_{n_j}\rangle \otimes k \) to Bob through a noiseless quantum channel and the message \( m \) through an unauthentic classical channel. At the reception, Bob carries through POVM measurements in bases \( Z \) and \( X \), defined for following sets

\[
E_Z = \{ E_0 = |0\rangle \langle 0|, E_1 = |1\rangle \langle 1| \}
\]

\[
E_X = \{ E_+ = |+\rangle \langle +|, E_- = |\mp\rangle \langle \mp| \}
\]

For \( j \)-th received qubit \( |\theta_{n_j}\rangle \), Bob carries through a measurement using set \( E_Z \) or \( E_X \), depending if \( j \)-th bit on \( y_a(n) \) is 0 or 1, respectively. Of this form, Bob considers that \( j \)-th bit of the tag of Alice is 0 when it gets the \( E_0 \) or \( E_+ \). If the exit will be \( E_1 \) or \( E_- \), Bob it considers that bit \( j \) of \( a(m, n) \) is 1. To the end of \( k \) measurements, Bob makes use of one sequence \( a_B(m, n) \) of \( k \) classic bits. Moreover, Bob also makes use of message received, denoted for \( m_B \) that it can be modified or not.

The next step to Bob is to calculate an established local label in the function hash and in sequence generated for the seed \( x_a \), getting

\[
a_B(m, n) = h(m_B) \oplus x_a(n)
\]
As the quantum channel is assumed to be perfect, Bob accept that the message is authentic if \( a_B(m, n) = a_B'(m, n) \). Otherwise it discards the received message. The statement above can be made because in the case where Eva does not intervene with quantum transmission of the label, Bob will get in the measurement the same sent tag for Alice, or either, \( a_B(m, n) = a(m, n) \). This is because the generator whose seed is \( y_k \), indicates in which of the Alice basis must create qubits, at the same time where which of the POVMs sets says the Bob it must choose for measure them. If the measurement it is made always in the same base where qubits were encoded Bob always interprets correctly the bit sent for Alice.

IV. COMMENTS SECURITY OF THE PROTOCOL WHEN THE BBS IS USED

It is clearly in this point that the security of the considered project depends on the security of the generating BBS. It is investigated as Eve makes use of quantum resources stops to predict such generator. First, the following result to the BBS is proven relative, where the parameters, \( p \) and \( q \) are primes numbers such that \( p \equiv 3 \mod 4 \) and \( q \equiv 3 \mod 4 \), and \( x_0 \) is a private seed chosen of \( Z_n^* \), having representation of \( l \) bits.

**Lemma 1:** Let \( b_{i+1}, \ldots, b_{i+k} \) a sequence of bits of the sequence generated for PRSG-BBS. The best probabilistic algorithm \( A_{BBS}(p, q, b_{i+1}, \ldots, b_{i+k}) \) to predict an entire sequence backwards (and forwards) needs for least \( k = l \) bit s so that

\[
\text{Prob}[A_{BBS}(p, q, b_{i+1}, \ldots, b_{i+k}) = b_i] = 1
\]

**Proof:** The proof is based on the remark that the problem is equivalent to find the root square shaped module \( N \). Either then \( A_{RQ}(p, q, x_{i+1}) \), the algorithm capable to decide the square shaped root of \( x_i \mod N \) where \( b_i = \text{parity}(x_i) \) in a quantum computer. The result follows for contradiction. Assume that such algorithm exists. Then should there exist a function \( f(b_{i+1}, \ldots, b_{i+k}) \) such that

\[
x_{i+1} = f(b_{i+1}, \ldots, b_{i+k}), \quad k < l
\]

and

\[
x_i = A_{RQ}(p, q, x_{i+1})
\]

\[
b_i = \text{parity}(x_i)
\]

But such function does not exist since the cardinality of the domain is \( 2^k \) is lesser that the cardinality of the co-domain that is \( 2^l \), what it is a contradiction searched.

The lemma above allows to establishes a proof of security against measurement attacks. We refer the reader to a detailed proof of this sort of security presented in [1].

We observe that information available to the eavesdropper must be less than that contained in the transmitted one, as an consequence of the process of its “acquisition”, so if he/she does not have quantum computation power (or equivalent) then the proposed protocol is computationally secure by the same reason it is secure relative to classical computers.

Let consider now that the attacker owns quantum computers. First we remark that quantum Fourier transform is a fundamental ingredient of quantum computer algorithms dedicated to factorization, discrete log calculation or more generally the problem of finding hidden subgroups. The crucial step of these algorithms (e.g. factoring) is that of phase estimation, which is implemented with a “inverse” quantum Fourier transform [9].

We assume also that for any sort of attack (e.g., individual or coherent) the sequence tap-wireded from the quantum channel differs from the sequence of quantum states transmitted through the quantum channel. That is, there is a probability \( p > 0 \) that a state entangled or read-out and resent differs from the actual original state transmitted. At this point it is clear that the eavesdropper will input his/her algorithm with a corrupted sequence. We conjecture that susceptibility of the phase estimation algorithm concerned to input deviation will works to keep the security of the protocol.

V. CONCLUSION

In this paper we show that the replacement of the BM PRSG by the BBS PRSG in the MA4 protocol does not alter its security. Some comments on the security were performed. An analysis of security in the case where the eavesdropper has quantum computation power based on the susceptibility of the phase estimation algorithm is on schedule.

REFERENCES


