UNIVERSITATEA "DUNĂREA DE JOS"
GALAȚI

FACULTATEA DE AUTOMATICĂ, CALCULATOARE,
INGINERIE ELECTRICĂ ȘI ELECTRONICĂ

TEZĂ DE DOCTORAT

SECURIZAREA SISTEMELOR INFORMATICE ȘI DE COMUNICAȚII PRIN CRIPTOGRAFIA CUANTICĂ

Conducător științific:
Profesor dr. ing. Adrian FILIPESCU

Doctorand:
ing. Cătălin ANGHEL

GALAȚI
2012

UNIVERSITATEA "DUNĂREA DE JOS"
GALAȚI

FACULTATEA DE AUTOMATICĂ, CALCULATOARE,
INGINERIE ELECTRICĂ ȘI ELECTRONICĂ

TEZĂ DE DOCTORAT

SECURIZAREA SISTEMELOR INFORMATICE ȘI DE COMUNICAȚII PRIN CRIPTOGRAFIA CUANTICĂ

Conducător științific:
Profesor dr. ing. Adrian FILIPESCU

Doctorand:
ing. Cătălin ANGHEL

GALAȚI
2012

Abstract

"While classical cryptography employs various mathematical techniques to restrict eavesdroppers from learning the contents of encrypted messages, in quantum mechanics the information is protected by the laws of physics. In classical cryptography an absolute security of information cannot be guaranteed. The Heisenberg uncertainty principle and quantum entanglement can be exploited in a system of secure communication, often referred to as 'quantum cryptography'. Quantum cryptography provides means for two parties to exchange an enciphering key over a private channel with complete security of communication." - *Artur Ekert*.

One of the main research directions of quantum cryptography is to discover new algorithms and to improve the existing ones. Security of quantum cryptographic algorithms is another research direction in quantum cryptography. Security growth of the quantum key distribution systems can be realized by detecting the eavesdropper quickly, precisely and without leaving any secret information in the hands of the enemy.

This thesis, present a quantum cryptographic communication protocol based on the final key obtained after the Secret Key Reconciliation and Privacy Amplification process of any quantum cryptographic algorithm named Base Selection and Transmission Synchronization protocol.

Also, this thesis presents a new method, named Quantum Bit Travel Time, to detect the enemy who try to tap the communication channel. The Quantum Bit Travel Time method can be implemented in every type of quantum key distribution scheme.

Finally, this thesis proposes a new quantum cryptographic communication algorithm based on Base Selection and Transmission Synchronization protocol with Quantum Bit Travel Time method of eavesdropper detection which can realize a perfectly secure communication between two computers, named Base Selection and Polarization Agreement.

Doctoral research results were presented in 6 articles published or accepted for publication in which: 1 paper in a journal indexed ISI (CNCSIS A), 2 papers in journals indexed BDI (CNCSIS B+), 2 papers in international conferences and 1 paper in electronic library of Cornell University Library from the USA.

Cuprins

ABSTRACT **1**

CUPRINS **3**

LISTA FIGURILOR **7**

LISTA TABELELOR **9**

LISTA ABREVIERILOR **10**

1. INTRODUCERE **11**

1.1 STRUCTURA TEZEI 11

1.2 ISTORIA CRIPTOGRAFIEI 13

1.3 INTRODUCERE ÎN CRIPTOGRAFIE 14

1.4 CRIPTOGRAFIA CLASICĂ 15

1.4.1 Securitatea criptării clasice *15*

1.4.2 Modelul criptării clasice *16*

1.5 CRIPTOGRAFIA MODERNĂ 17

1.6 CRIPTOGRAFIA CUANTICĂ 18

1.6.1 Principiul incertitudinii al lui Heisenberg *19*

1.6.2 Entanglementul cuantic – perechi EPR *20*

1.6.3 Qbiți – fotoni polarizați *20*

1.6.4 Distribuirea cheilor cuantice – QKD *23*

1.7 CONCLUZII 24

1.8 OBIECTIVELE TEZEI 25

2. SISTEME DE DISTRIBUIRE A CHEILOR CUANTICE **26**

2.1 INTRODUCERE 26

2.2 SISTEMUL DE DISTRIBUIRE A CHEILOR CUANTICE BB84 26

2.2.1 Pașii sistemului QKD BB84 ..... *26*

2.2.2 Detectarea inamicului *27*

2.2.3 Secret key reconciliation *27*

2.2.4 Privacy amplification *28*

2.2.5 Pseudocodul sistemului QKD BB84 ..... *28*

2.2.6 Diagrama funcțională a sistemului QKD BB84 ..... *30*

2.2.7 Exemplu teoretic *31*

2.3 SISTEMUL DE DISTRIBUIRE A CHEILOR CUANTICE B92 32

2.3.1 Pașii sistemului QKD B92 ..... *33*

2.3.2 Detectarea inamicului *33*

2.3.3 Secret key reconciliation *34*

2.3.4 Privacy amplification *34*
2.3.5 Pseudocodul sistemului QKD B92 ..... *35*
2.3.6 Diagrama funcțională a sistemului QKD B92 ..... *36*
2.4 SISTEMUL DE DISTRIBUIRE A CHEILOR CUANTICE E91 37
2.4.1 Pașii sistemului QKD E91 ..... *38*
2.4.2 Detectarea Inamicului *39*
2.4.3 Secret key reconciliation *40*
2.4.4 Privacy amplification *40*
2.4.5 Pseudocodul sistemului QKD E91 ..... *40*
2.4.6 Diagrama funcțională a sistemului QKD E91 ..... *42*
2.5 CONCLUZII 43
3. ANALIZA SECURITĂȚII SISTEMELOR DE DISTRIBUIRE A CHEILOR CUANTICE 44
3.1 INTRODUCERE 44
3.2 VULNERABILITĂȚILE SISTEMELOR QKD 44
3.2.1 Atacul de tip Intercept-Resend *45*
3.2.2 Atacul de tip Photon Number Splitting *46*
3.2.3 Atacul de tip Man-in-the-Middle *47*
3.2.4 Atacul de tip Quantum Cloning *47*
3.3 DETECTAREA ATACURILOR ASUPRA QKD 48
3.3.1 Metoda clasică *48*
3.3.2 Quantum Bit Error Rate – QBER *48*
3.3.3 Inegalitatea lui Bell *49*
3.4 CONCLUZII 49
4. PROIECTAREA PROTOCOLULUI BASE SELECTION AND TRANSMISSION SYNCHRONIZATION 50
4.1 INTRODUCERE 50
4.2 NECESITATEA PROTOCOLULUI BSTS 50
4.3 PREMISELE PROTOCOLULUI BSTS 51
4.4 PAȘII PROTOCOLULUI BSTS 53
4.5 EXEMPLU TEORETIC 55
4.6 PSEUDOCODUL PROTOCOLULUI BSTS 56
4.7 DIAGRAMA FUNCȚIONALĂ A PROTOCOLULUI BSTS 59
4.8 AVANTAJELE PROTOCOLULUI BSTS 60
4.9 CONCLUZII ȘI CONTRIBUȚII 61
4.9.1 Concluzii *61*
4.9.2 Contribuții *61*
5. PROIECTAREA METODEI QUANTUM BIT TRAVEL TIME DE DETECTARE A INAMICULUI ... 63

5.1 INTRODUCERE 63
5.2 NECESITATEA METODEI 63
5.3 PREMISELE METODEI QBTT 64
5.4 IMPLEMENTAREA METODEI QBTT ÎN SISTEMUL BB84 65
5.5 DETECTAREA INAMICULUI ÎN SISTEMUL BB84 CU METODA QBTT 65
5.6 PSEUDOCODUL UNUI SISTEM BB84 CU METODA QBTT 66
5.7 DIAGRAMA FUNCȚIONALĂ A SISTEMULUI BB84 CU METODA QBTT 68
5.8 IMPLEMENTAREA METODEI QBTT ÎN SISTEMUL B92 69
5.9 DETECTAREA INAMICULUI ÎN SISTEMUL B92 CU METODA QBTT 70
5.10 PSEUDOCODUL UNUI SISTEM B92 CU METODA QBTT 70
5.11 DIAGRAMA FUNCȚIONALĂ A SISTEMULUI B92 CU METODA QBTT 72
5.12 AVANTAJELE METODEI QBTT 73
5.13 CONCLUZII ȘI CONTRIBUȚII 73
5.13.1 Concluzii 73
5.13.2 Contribuții 74

6. PROIECTAREA PROTOCOLULUI CUANTIC BASE SELECTION AND POLARIZATION AGREEMENT 75

6.1 INTRODUCERE 75
6.2 PREMISELE PROTOCOLULUI BSPA 75
6.3 IMPLEMENTAREA PROTOCOLUL BSPA 77
6.4 PSEUDOCODUL PROTOCOLULUI BSPA 78
6.5 DIAGRAMA FUNCȚIONALĂ A PROTOCOLULUI BSPA 80
6.6 AVANTAJELE PROTOCOLULUI BSPA 81
6.7 CONCLUZII ȘI CONTRIBUȚII 81
6.7.1 Concluzii 81
6.7.2 Contribuții 82

7. SIMULAREA SISTEMELOR DE DISTRIBUIRE A CHEILOR CUANTICE 83

7.1 INTRODUCERE 83
7.2 SIMULAREA SISTEMULUI BB84 ÎN CONDIȚII IDEALE FĂRĂ INAMIC 84
7.2.1 Implementarea software 84
7.2.2 Implementarea hardware 84
7.2.3 Implementarea protocolului 84
7.2.4 Rezultate 85
7.2.5 Detalii de funcționare 86
7.3 SIMULAREA SISTEMULUI BB84 ÎN CONDIȚII IDEALE CU INAMIC 90
7.3.1 Implementarea software 90
7.3.2 Implementarea hardware 91

7.3.3 Implementarea protocolului *91*
7.3.4 Rezultate *92*
7.3.5 Detalii de funcționare *93*
7.4 SIMULAREA SISTEMULUI BB84 CU METODA QBTT 97
7.4.1 Implementarea software *97*
7.4.2 Implementarea hardware *97*
7.4.3 Implementarea protocolului *98*
7.4.4 Rezultate *98*
7.4.5 Detalii de funcționare *99*
7.5 SIMULAREA PROTOCOLULUI BSPA 101
7.5.1 Implementarea software *101*
7.5.2 Implementarea hardware *102*
7.5.3 Implementarea protocolului *102*
7.5.4 Rezultate *103*
7.5.5 Detalii de funcționare *104*
7.6 CONCLUZII ȘI CONTRIBUȚII 106
8. CONCLUZII, CONTRIBUȚII ȘI DIRECȚII DE CERCETARE 108
8.1 CONCLUZII 108
8.2 CONTRIBUȚII 111
8.3 DISEMINAREA REZULTATELOR CERCETĂRII 117
8.4 DIRECȚII DE CERCETARE 118
8.4.1 Mărirea dimensiunii cheii finale *118*
8.4.2 Securitatea QKD *118*
8.4.3 Mărirea distanței de transmisie *118*
8.4.4 Crearea și îmbunătățirea algoritmilor *119*
8.4.5 Utilizarea sateliților în QKD *119*
8.4.6 Repetoare cuantice *119*
8.4.7 Memorii cuantice *119*
8.4.8 Rețele QKD *120*
8.4.9 Calculatoare cuantice *120*
BIBLIOGRAFIE 121
ANEXA 1 - CODUL SURSĂ AL PROGRAMELOR 128
PROGRAMUL BB84 CU METODA QBTT 128
Emițător.cpp *128*
Receptor.cpp *133*
Inamic.cpp *137*

Lista figurilor

1.1. Un criptosistem clasic de comunicații 16
1.2. Polarizarea luminii 21
1.3. Polarizarea liniară 21
1.4. Polarizarea liniar-diagonală a fotonilor 22
1.5. Sistem de distribuire a cheilor cuantice de tip single-photon 24
1.6. Sistem de distribuire a cheilor cuantice de tip entanglement 24
2.1. Diagrama sistemului QKD BB84 30
2.2. Stările de polarizare non-ortogonale ale *emițătorului* 32
2.3. Stările de polarizare receptor 32
2.4. Interpretarea qbiților de către receptor 32
2.5. Diagrama sistemului QKD B92 36
2.6. Stările de polarizare non-ortogonale folosite în algoritmul E91 38
2.7. Diagrama sistemului QKD E91 42
3.1. Atac de tip Intercept-Resend 45
3.2. Atac de tip PNS 46
3.3. Atac de tip Man-in-the-Middle 47
3.4. Atac de tip Quantum Cloning 47
4.1. Diferența dintre cheia inițială și cea finală 51
4.2. Schema bloc a algoritmului de comunicații BSTS 53
4.3. Diagrama funcțională a protocolului BSTS 59
5.1. Diagrama funcțională a unui sistem BB84 cu metoda QBTT 68
5.2. Diagrama funcțională a unui sistem B92 cu metoda QBTT 72
6.1. Schema bloc a protocolului cuantic de comunicații BSPA 76
6.2. Diagrama funcțională a protocolului BSPA 80
7.1. Schema bloc a programului de simulare 83
7.2. Schema bloc a programului de simulare BB84-Ideal 84
7.3. Schema funcțională a programului de simulare BB84-Ideal 85
7.4. Detalii simulare BB84 în condiții ideale 86
7.5. Emițătorul - Bazele de polarizare, biții și qbiții pregătiți de emițător 87
7.6. Receptorul - Bazele de polarizare și qbiții receptorului 88
7.7. Emițătorul și Receptorul – obținerea cheii finale unice, sigure și secrete 89
7.8. Detalii 89

7.9. Atac de tip Intercept-Resend 90
7.10. Schema funcțională a simulării protocolului BB84 91
7.11. Detalii simulare BB84 în condiții ideale cu inamic 92
7.12. Emițătorul - Bazele de polarizare, biții și qbiții pregătiți de emițător 93
7.13. Inamicul - Bazele de polarizare și qbiții interceptați 94
7.14. Emițătorul și Receptorul – obținerea cheii brute 95
7.15. Emițătorul și Receptorul – obținerea cheii finale 96
7.16. Emițătorul, Receptorul și Inamicul 96
7.17. Schema funcțională a programului de simulare BB84 QBTT 98
7.18. Detalii simulare BB84 în condiții ideale cu inamic 99
7.19. PC - Emițătorul BB84 QBTT 100
7.20. PC - Inamic BB84 QBTT 100
7.21. PC - Receptor BB84 QBTT 101
7.22. Schema funcțională a programului de simulare BSPA 103
7.23. Detalii simulare BSPA 104
7.24. Emițătorul și receptorul în protocolul BSPA 104
7.25. Detalii de funcționare în protocolul BSPA 105
7.26. Detalii de transmisie în protocolul BSPA 105
7.27. Comparație între cheia inițială și cheia finală 106
7.27. Comparație între QBER ale celor patru programe de simulare 107
8.1. Comparație între cele patru programe de simulare 111
8.2. Diagrama funcțională a protocolului BSTS 112
8.3. Avantajele motodei QBTT 114
8.4. Diagrama functională a protocolului BSPA 115
8.5. Avantajele protocolului BSPA 116

Lista tabelelor

1.1. Timpi necesari pentru decriptare folosind forța brută 16
1.2. Polarizarea fotonilor - qbiți 22
2.1. Polarizarea fotonilor în BB84 26
2.2. Algoritmul BB84 – cazul ideal 31
2.3. Algoritmul BB84 – inamicul este prezent 31
2.4. Stările de polarizare folosite de *emițător* și *receptor* în B92 33
2.5. Pentru $|\psi A$ 37
2.6. Pentru $|\psi B$ 37
2.7. Pentru $|\psi C$ 37
4.1. Polarizarea fotonilor - qbiți 52
4.2. Selectarea bazelor în funcție de biții 1 și 2 din *cheia finală – base selection* 53
4.3. Stabilirea intervalului de timp al transmisiei - *transmission synchronization* 54
4.4. Stabilirea bazelor pentru polarizarea fiecărui bit 54
4.5. Biții din cheia finală 55
4.6. Biții 1 și 2 din *cheia finală – base selection* 55
4.7. Biții 3,4,5 și 6 din cheia finală – *transmission synchronization* 55
4.8. Selectarea bazelor de polarizare a biților 56
4.9. Bazele de polarizare a fotonilor 56
6.1. Polarizarea fotonilor - qbiți 75
6.2. Selectarea bazelor în funcție de biții 1 și 2 din *cheia finală – base selection* 77
6.3. Stabilirea bazelor pentru polarizarea fiecărui bit 77
7.1. Reprezentarea biților în programele de simulare 83
7.2. Simulare BB84 în condiții ideale 85
7.3. Simulare BB84 în condiții ideale cu inamic 92
7.4. Simulare BB84 în condiții ideale cu metoda QBTT 99
7.5. Simulare BSPA 103
7.6. Comparație între cheia inițială și cheia finală 106
7.7. Comparație între QBER ale celor patru programe de simulare 107
8.1. Comparație între cele patru programe de simulare 110
8.2. Avantajele metodei QBTT 114
8.3. Avantajele protocolului BSPA 116

Lista abrevierilor

OTP	one-time pad
RSA	Rivest, Shamir, Adleman
QKD	quantum key distribution
EPR	Einstein, Podolsky, Rosen
SPDC	spontaneous parametric down-conversion
BB84	Bennett și Brassard 1984
B92	Bennett 1992
E91	Ekert 1991
QBER	quantum bit error rate
Pr	probabilitatea
PNS	photon number splitting
MiM	man in the middle
PGQCM	pretty good quantum copying machine
BSTS	base selection and transmission synchronization
QBTT	quantum bit travel time
BSPA	base selection and polarization agreement

1. Introducere

1.1 Structura tezei

În capitolul 1, intitulat „Introducere", pe lângă partea de istorie a criptografiei și o scurtă introducere în criptografie, este făcută o analiză a securității sistemelor criptografice clasice și a celor moderne arătându-se că, cu o singură excepție, securitatea lor se bazează pe complexitatea matematică a calculelor. În funcție de dimensiunea cheii de criptare folosite, pentru criptanaliză pot fi necesari și câteva mii de ani pentru puterea de calcul a sistemelor informatice existente în zilele noastre. Singurul criptosistem demonstrat a fi sigur este one-time pad – OTP [71,72], dar pentru a fi considerat de securitate necondiționată trebuiesc respectate anumite cerințe legate de cheia de criptare și anume ca aceasta să fie de dimensiunea textului care se dorește a fi criptat, să fie utilizată doar o singură dată și să fie perfect sigură. De aici rezultă un paradox și anume că „înainte de a comunică în secret, trebuie să comunicăm în secret". Aici intervine criptografia cuantică care profitând de anumite fenomene ce au loc la nivel subatomic nu numai că face imposibilă interceptarea transmisiei dar poate și detecta dacă un atacator ascultă canalul de comunicații. Securitatea criptografiei cuantice nu se bazează pe presupusa complexitate a unei probleme matematice, ci pe principiile fizicii cuantice – mai exact, pe principiul incertitudinii al lui Heisenberg și pe entanglementul cuantic al particulelor. În continuare sunt prezentate cele două principii ale fizicii cuantice, sunt introduși qbiții și tipurile de polarizări ale acestora și modelele sistemelor de distribuire a cheilor cuantice de tip single-photon și de tip entanglement.

În capitolul 2, intitulat „Sisteme de distribuire a cheilor cuantice", sunt prezentate cele mai cunoscute și utilizate sisteme de distribuire a cheilor cuantice care vor fi folosite ulterior pentru dezvoltarea contribuțiilor proprii din cadrul tezei. Sunt descrise modurile de funcționare și implementare a schemelor de distribuire a cheilor cuantice BB84, B92 și E92, sunt prezentați pașii, metodele de detectare a inamicului, etapele distilării cheii finale, pseudocodul și schemele logice ale acestora.

În capitolul 3, intitulat „Analiza securității sistemelor de distribuire a cheilor cuantice", sunt prezentate cele mai cunoscute și utilizate metode de atac asupra

schemelor de distribuire a cheilor cuantice și cele mai cunoscute și utilizate metode de detectare a inamicului care vor fi folosite ulterior pentru dezvoltarea contribuțiilor proprii din cadrul tezei. Sunt prezentate și descrise atacurile de tip Intercept-Resend, Photon Number Splitting, Man-in-the-Middle și Quantum Cloning, precum și metodele de detectare a inamicului, cum ar fi Metoda Clasică, Quantum Bit Error Rate și cea care utilizează inegalitatea lui Bell.

În capitolul 4, intitulat „Proiectarea protocolul Base Selection and Transmission Synchronization", am propus un protocol cuantic de comunicații bidirecțional. Acest protocol criptografic cuantic de comunicații, folosește biții din cheia finală, obținută după etapele Secret key reconciliation și Privacy amplification, ale oricărui sistem de distribuire a cheilor cuantice și stabilește cu exactitate parametrii utilizați, de către cele două părți care vor să comunice, pentru realizarea transmisiei cuantice. Sunt prezentate etapele, pseudocodul și schema logică a noului protocol BSTS.

În capitolul 5, intitulat „Proiectarea metodei Quantum Bit Travel Time de detectare a inamicului", am propus o nouă metodă de detectare a unui inamic care interceptează un canal cuantic de comunicații și care poate fi implementată în orice sistem de distribuire a cheilor cuantice. Avantajele oferite de această nouă metodă sunt acelea că inamicul poate fi depistat în timpul transmisiei cuantice, după fiecare qbit recepționat și mai ales că poate fi depistat cu exactitate, fără dubii că ar putea fi confundat cu zgomotele de pe canalul cuantic sau cu erori ale dispozitivelor. Sunt prezentate în continuare modalitățile de implementare a metodei QBTT în sistemele BB84 și B92, schemele logice și pseudocodul.

În capitolul 6, intitulat „Proiectarea protocolului cuantic Base Selection and Polarization Agreement", am propus o variantă simplificată a protocolului BSTS care folosește pentru detectarea inamicului metoda QBTT. Protocolul BSTS cu metoda QBTT realizează o transmisie bidirecțională în totalitate cuantică, rezultând un algoritm cuantic de comunicații între un emițător și un receptor. Sunt prezentate în continuare modalitățile de implementare, pseudocodul și schema logică a noului protocol.

În capitolul 7, intitulat „Simularea sistemelor de distribuire a cheilor cuantice", am prezentat câteva programe de simulare a schimbului de chei cuantice scrise în C++. Sunt prezentate programul de simulare a sistemului BB84 în condiții ideale fără

inamic, programul de simulare a sistemului BB84 în condiții ideale cu inamic, programul de simulare a sistemului BB84 cu metoda QBTT și programul de simulare a unui sistem criptografic cuantic BSPA.

În capitolul 8, intitulat „Concluzii, contribuții și direcții de cercetare" sunt prezentate concluziile finale, contribuțiile originale din cadrul tezei, diseminarea rezultatelor cercetării precum și direcțiile viitoare de cercetare.

Rezultatele cercetărilor doctorale au fost prezentate în 6 articole publicate sau acceptate spre publicare din care : 1 lucrare într-o revistă indexată ISI cu factor de impact 0.913, 2 lucrări în reviste indexate BDI (CNCSIS B+), 2 lucrări la conferințe internaționale și 1 lucrare în biblioteca electronică Cornell University Library din SUA.

1.2 Istoria criptografiei

Informația a însemnat întotdeauna putere, prin urmare dorința de a o proteja, de a o face accesibilă doar unor elite, unor inițiați, s-a pus din cele mai vechi timpuri [39]. Primele texte cifrate descoperite până în prezent datează de circa 4000 de ani și provin din Egiptul Antic.

Există date privind utilizarea scrierii cifrate în Grecia antică încă din secolul al V-lea î.e.n. Pentru cifrare se folosea un baston în jurul căruia se înfășura, spirală lângă spirală, o panglică îngustă de piele, papirus sau pergament pe care, paralel cu axa, se scriau literele mesajului. După scriere, panglica era derulată, mesajul devenind indescifrabil. El putea fi reconstituit numai de către persoana care avea un baston identic cu cel utilizat la cifrare. În Roma antică, secretul informațiilor politice și militare se făcea utilizând scrierea secretă. Amintim cifrul lui Cesar, utilizat încă din timpul războiului galic.

Contribuția arabă ([Deavours] - Al Kadif) la dezvoltarea criptologiei, mai puțin cunoscută și mediatizată este de o remarcabilă importanță. David Kahn, unul dintre cei mai de seamă istoriografi ai domeniului, subliniază în cartea sa *The Codebreakers,* că, criptologia s-a născut în lumea arabă [39]. Primele trei secole ale civilizației islamice (700-1000 e.n.) au constituit, pe lângă o mare extindere politică și militară și o epocă de intense traduceri în limba arabă ale principalelor opere ale antichității grecești, romane, indiene, armene, ebraice, siriene. Unele cărți sursă erau scrise în limbi deja moarte, deci reprezentau în fapt texte cifrate, astfel încât

traducerea lor constituie primii pași în criptanaliză, deci originile criptologiei pot fi atribuite arabilor. Dezvoltările criptanalizei au fost mult sprijinite de studiile lingvistice ale limbii arabe. Arabii au preluat cunoștințele matematice ale civilizațiilor grecești și indiene. Arabii sunt cei care au introdus sistemul zecimal de numerotație și cifrele "arabe". Termenii "zero", "algoritm", "algebră" li se datorează tot lor. Însuși termenul de "cifru" ne vine de la arabi. El provine de la cuvântul arab "sifr" care reprezintă traducerea în arabă a cifrei zero din sanscrită. Conceptul de zero a fost deosebit de ambiguu la începuturile introducerii lui în Europa, în care sistemul de numerotație folosit era cel roman. De aceea se obișnuia să se spună despre cineva care vorbea neclar că vorbește ambiguu, ca un cifru. Acest înțeles de ambiguitate a unui mesaj poartă și azi denumirea de cifru. Prin urmare, putem concluziona că, încă din antichitate s-a încercat securizarea informației și a datelor transmise.

1.3 Introducere în criptografie

Criptografie = κρυπτός {kryptós} (ascuns) + γράφειν {gráfein} (a scrie)

Criptografia (cuvânt derivat din limba greacă a cuvintelor *kryptós* și *gráfein* reprezentând ***scriere ascunsă***) este știința care se ocupă cu studiul *codurilor* și *cifrurilor*. Un cifru este de fapt un algoritm criptografic care poate fi folosit pentru a transforma un mesaj clar (*text clar*) într-un mesaj indescifrabil (*text cifrat*). Acest proces de transformare se numește *criptare* iar procesul invers se numește *decriptare*. Textul cifrat poate fi transmis ulterior prin orice canal de comunicații fără a ne face griji că informații sensibile ar putea ajunge în mâinile inamicilor.

Inițial, securitatea unui cifru depindea de faptul că inamicul nu cunoștea algoritmul de criptare folosit, dar pe măsură ce criptografia a evoluat, securitatea cifrului s-a bazat pe utilizarea unei chei secrete care se poate extrage din textul cifrat. Până la jumătatea secolului XX, nu a fost demonstrat faptul că un anumit cifru nu poate fi spart, ba chiar întreaga istorie a criptografiei este plină de relatări în care anumit cifru era spart iar ulterior erau creați alți algoritmi care la rândul lor erau sparți. Știința care se ocupă cu spargerea cifrurilor se numește *criptanaliza*. Sistemul format dintr-un algoritm de criptare și o cheie de criptare se numește *criptosistem*.

1.4 Criptografia clasică

Toate criptosistemele pot fi împărțite în două tipuri: criptosisteme simetrice numite și clasice sau convenționale și criptosisteme asimetrice numite și moderne. Criptosistemele simetrice, sau cu cheie secretă, sunt acele criptosisteme în care numai emițătorul și receptorul cunosc cheia secretă pe care o aplică la fiecare criptare sau decriptare. Criptosistemele asimetrice, sau cu cheie publică, se bazează pe perechi de chei. Una din chei (cheia publică) este folosită la criptare, iar cealaltă (cheia privată) este folosită la decriptare.

În criptografia clasică mesajul clar, numit și text clar, este convertit într-o secvență aparent aleatoare și fără sens, numită text cifrat. Procesul de criptare presupune un algoritm de criptare și o cheie de criptare. Această cheie este o valoare independentă de textul care se dorește a fi criptat. Odată produs, textul criptat trebuie transmis destinatarului. La recepție acest text criptat trebuie transformat în textul original folosind un algoritm de decriptare bazat pe aceeași cheie folosită la criptare.

1.4.1 Securitatea criptării clasice

Securitatea criptării convenționale depinde de două aspecte esențiale: algoritmul de criptare și cheia de criptare [4]. Algoritmul de criptare, care trebuie să fie destul de puternic pentru a face imposibilă o decriptare numai pe baza textului criptat. Cheia de criptare trebuie să fie destul de mare pentru a asigura o criptare puternică și mai ales trebuie să fie secretă, fără a păstra secret algoritmul folosit, ci numai a cheii de criptare.

Există două cerințe esențiale care trebuie să le îndeplinească un algoritm de criptare :

1) Costul spargerii codului să depășească valoarea informației criptate;

2) Timpul necesar spargerii codului să depășească timpul de viață al informației.

Un algoritm de criptare care satisface aceste două cerințe este numit algoritm cu securitate computațională.

Prezentăm în tabelul 1.1, cât timp este necesar pentru a decripta un text cifrat, folosind metoda brute force, pentru diferite dimensiuni ale cheii de criptare.

Dimensiunea cheii (biți)	Numărul de chei posibile	Timpul necesar pentru 1 decriptare / μs	Timpul necesar pentru 10^6 decriptări / μs
64	$2^{64} = 1.8 \times 10^{19}$	2^{63} μs = 2.9×10^{5} ani	106 zile
128	$2^{128} = 3.4 \times 10^{38}$	2^{127} μs = 5.4×10^{24} ani	5.4×10^{18} ani
256	$2^{256} = 1.1 \times 10^{77}$	2^{255} μs = 1.8×10^{63} ani	1.8×10^{57} ani
512	$2^{512} = 1.3 \times 10^{154}$	2^{511} μs = 6.7×10^{153} ani	6.7×10^{147} ani

Tabelul 1.1. Timpi necesari pentru decriptare folosind forța brută

Un algoritm de criptare este de securitate necondiționată dacă textul criptat generat de acesta nu conține destulă informație pentru a extrage textul original, indiferent de volumul de text decriptat care se află în posesia atacatorului. De asemenea, nu contează puterea de calcul și timpul de care dispune inamicul, mesajul nu poate fi decriptat deoarece informația necesară nu este acolo.

Cu excepția unui algoritm numit one-time pad, propus de Gilbert Vernam [71, 72] în 1920, nu există algoritm de criptare care să fie de o securitate necondiționată. Securitatea acestui algoritm a fost demonstrată în 1949 de către Claude Shannon, condițiile fiind ca, cheia de criptare să fie de aceeași lungime cu textul clar, să fie secretă și să nu fie folosită decât o singură dată [62]. Algoritmul one-time pad a fost implementat pentru transmiterea de informații sensibile, dar, marea problemă rămâne distribuirea cheilor de criptare care trebuiesc schimbate la fiecare utilizare și sunt foarte mari.

1.4.2 Modelul criptării clasice

Un model de criptosistem simetric (clasic) este prezentat în figura 1.1.

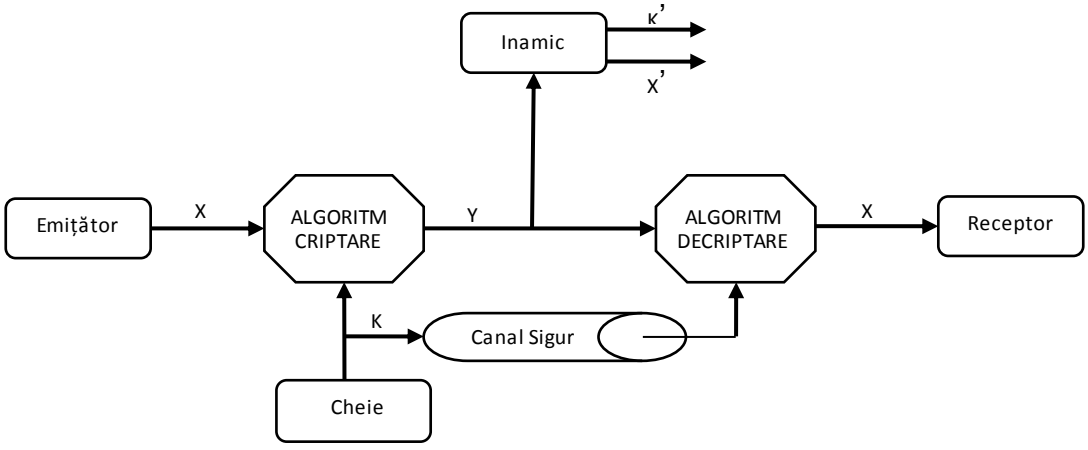

Figura 1.1. Un criptosistem clasic de comunicații

O sursă – *emițătorul,* produce un mesaj de tip text simplu (un șir de caractere), $X = [X_1,X_2,\ldots,X_M]$. Cele M elemente din X sunt niște litere ale unui alfabet finit. Pentru criptare trebuie generată o cheie de forma $K = [K_1,K_2,\ldots,K_J]$. Dacă cheia este generată de sursa mesajului aceasta trebuie transmisă destinatarului – *receptorului* printr-un canal de comunicație sigur. O alternativă este ca o a treia parte să genereze cheia care să fie transmisă celor două părți folosind o metodă cu grad înalt de securitate.

Cu mesajul X și cheia de criptare K ca și intrare, algoritmul de criptare formează textul criptat $Y = [Y_1,Y_2,\ldots,Y_N]$. Astfel putem scrie:

$$Y = E_K(X) \tag{1.1}$$

Această notație indică faptul că Y este produs folosind algoritmul E pe baza textului X cu o funcție specifică determinată de valoarea cheii K.

Receptorul, aflat în posesia cheii, poate inversa procesul de transformare:

$$X = D_K(Y) \tag{1.2}$$

unde D este algoritmul de decriptare.

Inamicul, observând Y și presupunând că cunoaște algoritmul de criptare E și decriptare D, va încerca să reconstituie X sau K sau ambele.

Procesul prin care se încearcă descoperirea lui X sau K, adică a textului necriptat, respectiv a cheii de criptare, se numește criptanaliză.

1.5 Criptografia modernă

Toate criptosistemele clasice moderne pot fi împărțite în două tipuri: criptosisteme simetrice și criptosisteme asimetrice. Criptosistemele simetrice, sau cu cheie secretă, sunt acele criptosisteme în care numai emițătorul și receptorul cunosc cheia secretă pe care o aplică la fiecare criptare sau decriptare. Criptosistemele asimetrice, sau cu cheie publică, se bazează pe perechi de chei. Una din chei (cheia publică) este folosită la criptare, iar cealaltă (cheia privată) este folosită la decriptare.

Un criptosistem simetric demonstrat a fi sigur este one-time pad, propus de Gilbert Vernam [71, 72] în 1920. Securitatea acestui algoritm a fost demonstrată în 1949 de către Claude Shannon, condițiile fiind ca, cheia de criptare să fie de aceeași lungime cu textul clar, să fie secretă și să nu fie folosită decât o singură dată [62].

Algoritmul one-time pad a fost implementat pentru transmiterea de informații sensibile dar, marea problemă rămâne distribuirea cheilor de criptare care trebuiesc schimbate la fiecare utilizare și sunt foarte mari.

Un criptosistem asimetric care deocamdată este considerat a fi sigur poate fi algoritmul RSA [59], implementat în 1978 de către Ronald Rivest, Adi Shamir și Leonard Adleman. Algoritmul RSA este folosit în prezent pentru securizarea comunicațiilor din internet, a tranzacțiilor bancare sau a comerțului electronic. Securitatea lui se bazează pe complexitatea matematică pe care o impune factorizarea numerelor prime.

Deci, criptosistemele cu chei publice suplinesc dezavantajul major al celor cu cheie secretă datorită faptului că nu mai este necesar schimbul de chei. Totuși, criptosistemele RSA au marele inconvenient că se bazează pe complexitatea matematică a calculelor, în funcție de dimensiunea cheii folosite pentru criptanaliză pot fi necesari și câteva mii de ani. Având în vedere faptul că, încă din 1985, David Deutch a descris principiile de funcționare ale unui calculator cuantic [22] – un supercalculator cu o putere de calcul extraordinar de mare care funcționează pe principiile fizicii cuantice, putem spune că criptosistemele cu chei publice nu sunt 100% sigure. În concluzie, singurul criptosistem absolut sigur rămâne one-time pad. Problema schimbului de chei poate fi rezolvată printr-un sistem de distribuire a cheilor cuantice (QKD – Quantum Key Distribution).

1.6 Criptografia cuantică

Criptografia cuantică ne pune la dispoziție noi metode de securizare a comunicațiilor. Spre deosebire de criptografia clasică, care implică diverși algoritmi matematici pentru a securiza informația, criptografia cuantică se concentrează pe suportul fizic al informației.

Procesul de transmitere sau stocare a informației este realizat prin intermediul unui suport fizic, spre exemplu fotonii transmiși prin fibră optică sau electronii din curentul electric. Securizarea comunicațiilor poate fi privită ca securizarea suportului fizic al purtătorului informației – în cazul nostru fotonii din fibra optică. Astfel, cum și ce poate un atacator afla depinde exclusiv de legile fizicii.

Folosind principiile fizicii cuantice, putem realiza și implementa un sistem de comunicații care va detecta întotdeauna orice încercare de atac [10], datorită faptului

că orice încercare de „măsurare" a unui purtător cuantic de informație va modifica particula purtătoare și va lăsa „urme".

În concluzie, criptografia cuantică, profitând de anumite fenomene ce au loc la nivel subatomic, nu numai că face imposibilă interceptarea transmisiei, dar poate de asemenea detecta dacă un atacator „ascultă" canalul de comunicații.

Spre deosebire de criptografia clasică, securitatea criptografiei cuantice nu se bazează pe presupusa complexitate a unei probleme matematice, ci pe principiile fizicii cuantice – mai exact, pe *Principiul incertitudinii al lui Heisenberg* [33] și pe *Entanglementul cuantic al particulelor* [26].

1.6.1 Principiul incertitudinii al lui Heisenberg

În 1927, Werner Heisenberg (1901-1976) a stabilit că este imposibil să măsurăm exact atât poziția unei particule, cât și impulsul ei. Cu cât determinăm mai precis pe una dintre ele, cu atât mai puțin o știm pe cealaltă [33]. Cele două variabile, poziția și impulsul, se numesc *variabile cuantice* sau *variabile legate*. Acest principiu este denumit *Principiul incertitudinii al lui Heisenberg* și este o proprietate fundamentală a mecanicii cuantice.

Deci, dacă măsurăm o anumită proprietate cuantică, vom modifica într-o anumită măsură o altă proprietate cuantică.

Principiul incertitudinii al lui Heisenberg poate fi enunțat după cum urmează :

Pentru oricare două observabile cuantice A și B avem :

$$\langle(\Delta A)^2\rangle\langle(\Delta B)^2\rangle \geq \frac{1}{4}\|\langle[A,B]\rangle\|^2, \tag{1.3}$$

unde

$$\Delta A = A - \langle A\rangle \;\; ; \;\; \Delta B = B - \langle B\rangle \tag{1.4}$$

și

$$[A,B] = AB - BA \tag{1.5}$$

Astfel, $\langle(\Delta A)^2\rangle$ și $\langle(\Delta B)^2\rangle$ sunt deviațiile standard care măsoară incertitudinea observabilelor A și B. Pentru observabilele A și B, pentru care $[A,B] \neq 0$, reducerea

incertitudinii $\langle(\Delta A)^2\rangle$ al observabilei A forțează creșterea incertitudinii $\langle(\Delta B)^2\rangle$ al observabilei B și invers.

1.6.2 Entanglementul cuantic – perechi EPR

În 1935, Einstein, Podolsky și Rosen (EPR), prezintă o lucrare [25] prin care aduc o provocare fundamentelor mecanicii cuantice arătând un paradox. Există perechi de particule cuantice, numite *perechi EPR* sau *particule entanglate*, care sunt separate spațial dar care sunt legate între ele de așa natură încât stările lor cuantice sunt interconectate, în sensul că, măsurând starea cuantică a uneia dintre particule vom ști automat starea cuantică a celeilalte particule.

În mecanica cuantică, spinul unei particule cuantice este nedeterminat, atâta vreme cât nu se intervine fizic pentru a-l măsura. Măsurarea stării cuantice a unui număr de particule duce la un rezultat impredictibil, deoarece jumătate din măsurări vor fi cu spinul în sus și jumătate vor fi cu spinul în jos.

Dacă aceleași măsurări le facem cu particule *entanglate*, rezultatele vor fi total predictibile. Spre exemplu, putem crea o pereche de fotoni *entanglați*, polarizați circular (spin-up, spin-down), a căror stare ***entanglată*** să fie reprezentată prin :

$$|\psi\rangle = \frac{1}{\sqrt{2}}(|\uparrow\downarrow\rangle - |\downarrow\uparrow\rangle) = \frac{1}{\sqrt{2}}(|\uparrow\rangle_1|\downarrow\rangle_2 - |\downarrow\rangle_1|\uparrow\rangle_2) \quad (1.6)$$

Astfel, dacă unul dintre fotoni este măsurat ca fiind în starea $|\downarrow\rangle$, celălalt, când va fi măsurat, va fi găsit sigur în starea $|\uparrow\rangle$ și invers.

Fotonii *entanglați* sunt obținuți printr-un proces numit *spontaneous parametric down-conversion - SPDC* [76]. Astfel, un foton polarizat trece printr-un cristal nonliniar [46] și este divizat în doi fotoni *entanglați* care vor avea stări de polarizare opuse (după măsurare). Particulele cuantice utilizate în criptografia cuantică sunt fotonii polarizați, numiți *qbiți*.

1.6.3 Qbiți – fotoni polarizați

Undele electromagnetice precum lumina pot prezenta fenomenul de polarizare, în care direcția vibrațiilor câmpului electric este constant și poate fi liniar

(orizontal - 0^o, vertical - 90^o, diagonal - 45^o, diagonal - 135^o) sau circular (spin-up, spin-down) – figura 1.2.

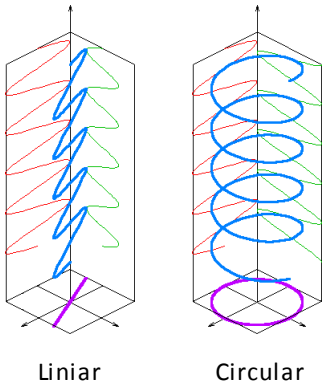

Figura 1.2. Polarizarea luminii

Conform mecanicii cuantice, lumina se propagă sub forma unor particule discrete numite fotoni. Un foton este o particulă fără masă, dar poartă energie, impuls și moment cinetic.

Polarizarea fotonilor se realizează cu ajutorul unor dispozitive speciale, numite filtre de polarizare, care permit trecerea numai a particulelor polarizate într-o anumită direcție. Un foton poate trece printr-un filtru de polarizare, dar dacă fotonul are o altă direcție de polarizare decât filtrul, atunci acel foton își va schimba polarizarea după direcția filtrului, figura 1.3.

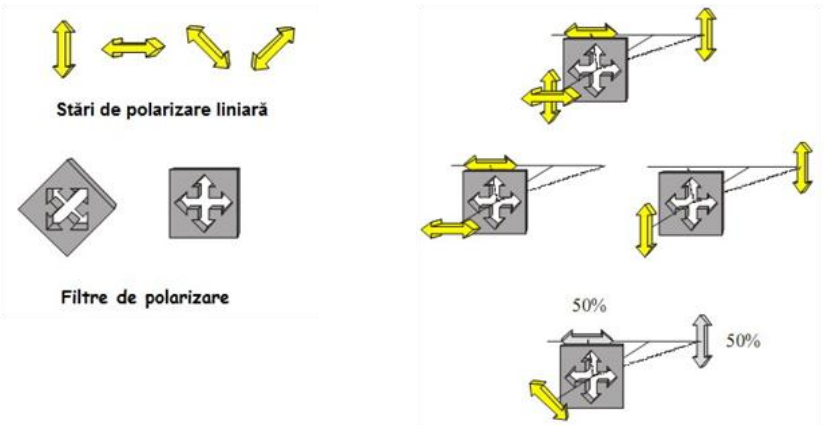

Figura 1.3. Polarizarea liniară

În criptografia cuantică considerăm că un foton poate fi polarizat *liniar* (0^o,90^o), *diagonal* (45^o,135^o) sau *circular* (stânga - spinL, dreapta - spinR), transformându-l prin polarizare în *qbit* [58]. În algoritmii de distribuire a cheilor cuantice, se folosesc în

general, pentru polarizarea fotonilor, două dintre cele trei tipuri de polarizare existente, figura 1.4.

Figura 1.4. Polarizarea liniar-diagonală a fotonilor

Fotonii polarizați liniar-diagonal, liniar-circular sau diagonal-circular sunt transformați în *qbiți* a căror stări sunt cunoscute ca fiind *stări legate* iar legile fizicii cuantice și anume Principiul incertitudinii al lui Heisenberg spune că este imposibil de măsurat simultan stările oricărei perechi de *variabile legate;* cu alte cuvinte, dacă *inamicul* încearcă să „măsoare" un foton polarizat orizontal, folosind o metoda de „măsurare" a fotonilor polarizați diagonal, acel foton își schimbă polarizarea din orizontală în diagonală.

În tabelul 1.2 vom stabili o convenție de notare pentru codificarea biților prin fotoni polarizați – qbiți.

Baza	**L**(Liniar)	**D**(Diagonal)	**C**(Circular)	**C**(Circular)	**L**(Liniar)	**D**(Diagonal)
Polarizare	0^o	45^o	spinL	spinR	90^o	135^o
Qbit	↔	↗	↺	↻	↕	↘
Bit	0	0	0	1	1	1

Tabelul 1.2. Polarizarea fotonilor - qbiți

Având în vedere cele menționate, rezultă că, folosind principiile fizicii cuantice, putem crea un sistem prin care se poate realiza *distribuirea cheilor cuantice* (QKD – Quantum Key Distribution) în deplină siguranță.

1.6.4 Distribuirea cheilor cuantice – QKD

Prin distribuirea cheilor cuantice [13,14,16,19,24,26,36,40], două entități, *emițătorul* și *receptorul*, stabilesc în comun o cheie unică și sigură care poate fi folosită împreună cu un algoritm de criptare sigur cum ar fi *one-time pad* [71,72].

Distribuirea cheilor cuantice utilizează avantajul unor fenomene ce au loc la nivel subatomic, astfel încât, orice încercare de interceptare a qbiților, nu numai că eșuează dar și alertează că s-a produs o interceptare.

O schemă clasică de distribuire a cheilor cuantice utilizează două canale de comunicații, unul clasic și unul cuantic și poate avea următoarele etape principale :

1. *Emițătorul* și *receptorul* generează secvențe de biți aleatoare și independente;
2. *Emițătorul* și *receptorul* folosesc un protocol de distribuire a cheilor cuantice pentru a compara secvențele de biți și pentru a stabili în comun o cheie unică și secretă;
3. *Emițătorul* și *receptorul* execută o procedură de corectare a erorilor numită *Secret key reconciliation* [11].
4. *Emițătorul* și *receptorul* apreciază (funcție de procentul de erori) dacă transmisia a fost interceptată de către *inamic*;
5. *Emițătorul* și *receptorul*, comunică printr-un canal public și execută o procedură numită *Privacy amplification* [9,11];
6. *Cheia finală, secretă, unică și sigură* este obținută.

Există două tipuri de scheme de distribuire a cheilor cuantice. Una folosește un generator care emite câte un singur foton și se bazează pe *principiul incertitudinii al lui Heisenberg* (*single-photon*) [13,14,16,19], în care se încadrează algoritmii BB84 și B92. Cealaltă folosește un generator care emite câte o pereche de fotoni entanglați și se bazează pe *entanglementul cuantic* al fotonilor [24,26,36,40], în care se încadrează algoritmul E91.

În cazul schemelor de tipul *single-photon* [13,14,16,19], fiecare bit din cheia transmisă, corespunde unei stări particulare a particulei purtătoare, cum ar fi fotonii polarizați – qbiți. Emițătorul cheii trebuie să stabilească o secvență de polarizare a fotonilor, cu care biții din cheie vor fi polarizați, după care vor fi transmiși printr-o fibră optică. Pentru a obține cheia, care este formată dintr-o secvență qbiți, *Receptorul* trebuie să facă o serie de măsurători, cu ajutorul unor filtre speciale, pentru a

determina polarizarea fotonilor. Detectarea inamicului se realizează prin metode specifice fiecărui algoritm în parte.

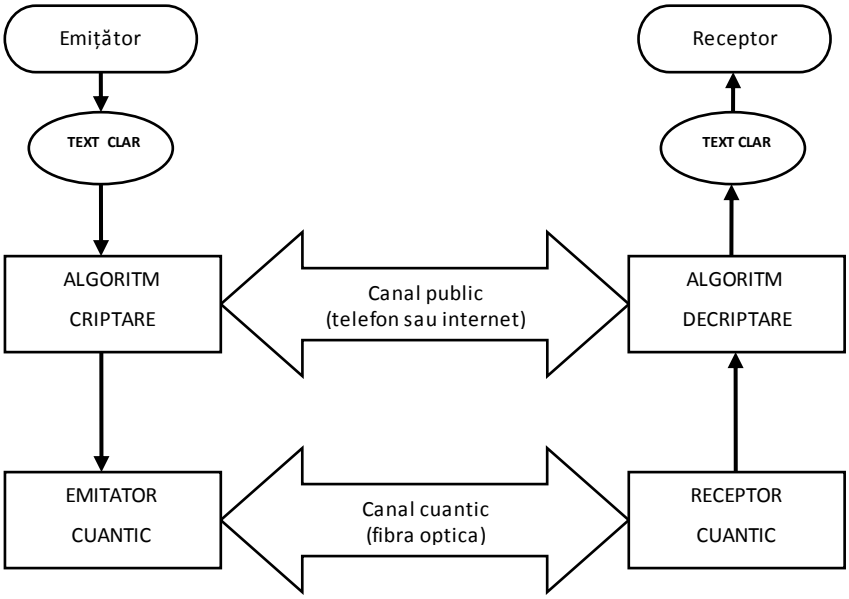

Figura 1.5. Sistem de distribuire a cheilor cuantice de tip single-photon

În cazul schemelor de tipul *entanglement* [24,26,36,40], un generator cuantic, creează perechi entanglate de fotoni polarizați, care sunt separați și transmiși simultan către cele două părți care comunică. Cu ajutorul filtrelor speciale fotonii polarizați sunt citiți și transformați în biți. Detectarea inamicului este determinată folosind *inegalitatea lui Bell* [8].

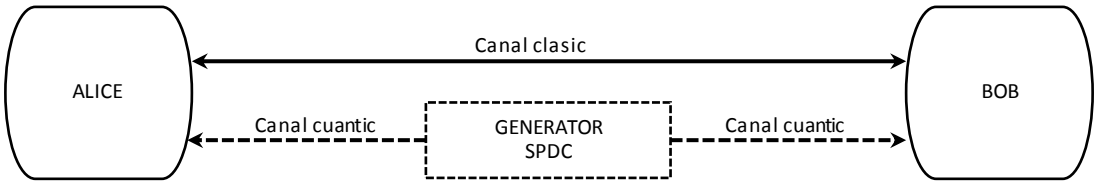

Figura 1.6. Sistem de distribuire a cheilor cuantice de tip entanglement

1.7 Concluzii

Am prezentat în acest capitol criptosistemele clasice și moderne. Dintre criptosisteme moderne asimetrice cel mai sigur este considerat a fi deocamdată

algoritmul RSA, dar securitatea acestuia se bazează pe complexitatea matematică pe care o impune factorizarea numerelor prime iar în funcție de dimensiunea cheii folosite pentru criptanaliză pot fi necesari și câteva mii de ani, la puterea actuală de calcul a sistemelor informatice existente.

Singurul criptosistem considerat a fi absolut sigur este one-time pad [71,72], dar rămâne o singură mare problemă și anume a cheii de criptare care trebuie să fie folosită o singură dată, să fie de dimensiunea textului criptat și să fie absolut secretă. Aici intervine criptografia cuantică care, folosind fotonii polarizați pentru transmiterea informației și bazându-se pe principiile fizicii cuantice – principiul incertitudinii al lui Heisenberg și entanglementul cuantic al particulelor, realizează în condiții de securitate necondiționată un schimb de chei cuantice care nu poate fi interceptat de către inamic.

În concluzie, putem spune că utilizând algoritmul one-time pad și folosind un sistem de distribuire a cheilor cuantice - QKD pentru obținerea cheii secrete rezultă un sistem criptografic de securitate necondiționată.

1.8 Obiectivele tezei

Principalul obiectiv al acestei teze este securizarea comunicațiilor ce au loc între sistemele informatice și de comunicații prin criptografie cuantică.

Securitatea criptografiei cuantice presupune securitatea sistemelor de distribuire a cheilor cuantice.

Securitatea sistemelor de distribuire a cheilor cuantice poate fi mărită prin îmbunătățirea schemelor existente sau prin descoperirea unor noi scheme mai performante și mai sigure.

Întărirea securității acestor sisteme de distribuire a cheilor cuantice se poate realiza prin descoperirea unor noi metode de detectare a inamicului, metode mai eficiente, mai sigure și mai rapide.

Pentru atingerea obiectivului principal mi-am propus proiectarea și implementarea unui protocol cuantic de comunicații, protocol ce va realiza o comunicare bidirecțională în totalitate cuantică între două sisteme informatice și nu în ultimul rând proiectarea și implementarea unei metode de detectare a interceptării care să poată depista prezența oricărui inamic, indiferent de metoda de atac folosită de către acesta.

2. Sisteme de distribuire a cheilor cuantice

2.1 Introducere

În continuare, sunt prezentate cele mai cunoscute și utilizate sisteme de distribuire a cheilor cuantice BB84 și B92 care utilizează un singur foton și E91 care utilizează o pereche de fotoni entanglați, obținuți prin procedeul *spontaneous parametric down-conversion – SPDC* [76]. Sunt descrise modurile de funcționare și implementare, sunt prezentați pașii, metodele de detectare a inamicului, etapele distilării cheii finale, pseudocodul și schemele logice ale celor trei scheme de distribuire a cheilor cuantice amintite.

2.2 Sistemul de distribuire a cheilor cuantice BB84

Sistemul de distribuire a cheilor cuantice (QKD) BB84, este primul algoritm cuantic [1,10,21,61] și a fost propus în 1984 de către Charles Bennett și Gilles Brassard. Conform acestuia, două entități, *emițătorul* și *receptorul* stabilesc în secret o cheie unică, comună și secretă, folosind fotoni polarizați – *qbiți* [63,66-68].

Pentru implementarea acestui sistem de distribuire a cheilor cuantice, vom folosi pentru polarizarea fotonilor bazele de polarizare liniară (L) și diagonală (D) iar pentru codificarea biților convenția din tabelul 2.1.

Baza	**L**	**D**	**L**	**D**
Stare	0^o	45^o	90^o	135^o
Qbit	→	↗	↑	↖
Bit	0	0	1	1

Tabelul 2.1. Polarizarea fotonilor în BB84

2.2.1 Pașii sistemului QKD BB84

1. *Emițătorul* generează o secvență aleatoare de biți – notată "***s***".
2. *Emițătorul* alege aleator ce bază de polarizare să aplice fiecărui foton din "***s***" (liniară "L" sau diagonală "D"). Notăm "***b***" secvența de polarizare.
3. *Emițătorul*, folosind un echipament special, creează o secvență "***p***" de fotoni polarizați – qbiți, a căror direcție de polarizare reprezintă biții din "***s***".

4. *Emițătorul* transmite qbiții din "***p***" prin fibră optică către *receptor*.
5. *Receptorul*, pentru fiecare qbit recepționat – secvența "***p'***", alege aleator câte o bază de polarizare (liniară "L" sau diagonală "D"). Notăm cu "***b'***" secvența bazelor de polarizare aleasă.
6. *Receptorul*, măsoară fiecare qbit recepționat respectând baza de polarizare aleasă la pct. 5, rezultând o secvență de biți "***s'***".
7. *Emițătorul* îi transmite printr-un canal public *receptorului*, ce bază de polarizare a ales pentru fiecare bit în parte. La rândul sau *receptorul* îi comunică *emițătorului* unde a făcut aceeași alegere a bazei de polarizare. Biții pentru care cei doi nu au avut aceeași bază de polarizare sunt eliminați din "***s***" și "***s'***".

2.2.2 Detectarea inamicului

Pentru qbitul cu numărul **n**, bitului ***s***[n] îi va corespunde o bază de polarizare ***b***[n] iar bitului ***s'***[n] îi va corespunde o bază ***b'***[n] .

Dacă ***b'***[n] = ***b***[n] va implica că ***s'***[n] = ***s***[n].

În cazul în care un inamic a încercat să citească fotonul purtător al lui ***s***[n], atunci chiar dacă cele două baze alese de *receptor* și *emițător* sunt identice (***b'***[n] = ***b***[n]), vom avea ***s'***[n] ≠ ***s***[n]. Aceasta le va permite *emițătorului* și *receptorului* să detecteze prezența inamicului și să reprogrameze transmisia.

Algoritmul BB84 este incomplet datorită faptului că, chiar dacă *inamicul* este prezent sau nu, vor exista erori generate de zgomote, în secvența de biți primită de *receptor*.

Pasul final al algoritmului BB84 constă într-o comparație dintre cele două secvențe de biți, deținute de *emițător* și *receptor* după codificare și decodificare. Acesta cuprinde două etape: *Secret key reconciliation* [11] și *Privacy amplification* [9,11].

2.2.3 Secret key reconciliation

Etapa *Secret key reconciliation* [12] este o procedură de corectare a erorilor din cheia brută, care elimină :

- erorile generate de alegerea diferită a bazelor
- erorile generate de *Inamic*
- erorile generate de zgomotele de pe canalul cuantic

Etapa *Secret key reconciliation* realizează o căutare binară, interactivă a erorilor. *Emițătorul* și *receptorul* împart secvența de biți rămasă (cheia brută) în blocuri de biți și vor compara paritatea fiecărui bloc. În cazul în care paritatea unui bloc de biți diferă, *emițătorul* și *receptorul* vor împărți blocul respectiv în blocuri mai mici și vor compara paritatea lor.

Acest proces va fi repetat până când bitul care diferă va fi descoperit și eliminat. Comunicațiile din această etapă se vor realiza pe un canal public nesecurizat la sfârșitul cărei se va obține cheia secretă.

2.2.4 Privacy amplification

Având în vedere faptul că, în etapa anterioară, comunicația s-a realizat pe un canal nesecurizat, există posibilitatea ca *inamicul* să dețină informații sensibile despre cheia secretă. Pentru a stabili o cheie perfect sigură, *emițătorului* și *receptorului* trebuie să mai realizeze o etapă : *Privacy amplification.* Această etapă constă într-o permutare a biților din cheia secretă și eliminarea unui subset de biți, care va fi realizată de către *emițător* și *receptor*.

La finalul acestei etape avem certitudinea că *emițătorul* și *receptorul*, dețin aceeași cheie unică și secretă care nu este cunoscută de *inamic*.

2.2.5 Pseudocodul sistemului QKD BB84

Pentru sistemul de distribuire a cheilor cuantice BB84, pseudocodul este următorul :

Pasul 1 – comunicarea pe canalul cuantic
Emițător:
generează aleatoriu dintre (0,1) șirul ***s***
FOR fiecare bit din ***s***
    alege aleatoriu dintre ("L", "D") rezultând baza ***b***[i]
ENDFOR
FOR fiecare bit din ***s***
    generează un foton
    IF ***s***[i] = 1 și ***b***[i] = L

THEN *polarizează* fotonul în starea (90^o) rezultă un qbit ***p***[i] = ↑
IF ***s***[i] = 0 și ***b***[i] = L
THEN *polarizează* fotonul în starea (0^o) rezultă un qbit ***p***[i] = →
IF ***s***[i] = 1 și ***b***[i] = D
THEN *polarizează* fotonul în starea (135^o) rezultă un qbit ***p***[i] = ↖
IF ***s***[i] = 0 și ***b***[i] = D
THEN *polarizează* fotonul în starea (45^o) rezultă un qbit ***p***[i] = ↗
trimite qbitul ***p***[i] către ***Receptor***
ENDFOR

Receptor:
FOR fiecare qbit ***p'***[i] recepționat
generează aleatoriu dintre ("L", "D") rezultând baza ***b'***[i]
măsoară qbitul ***p'***[i] în baza ***b'*** [i] rezultă bitul ***s'***[i]
ENDFOR

Pasul 2 – comunicarea pe canalul clasic

Receptor:
FOR fiecare bit ***s'***[i]
transmite baza ***b'***[i] către ***Emițător***
ENDFOR

Emițător:
FOR fiecare bit ***s***[i]
transmite baza ***b***[i] către ***Receptor***
ENDFOR

Pasul 3 – Bases reconciliation

Receptor:
FOR fiecare bit ***s'***[i]
IF baza ***b'***[i] ≠ ***b***[i]
elimină ***s'***[i] din șirul ***s'***
ENDIF
ENDFOR

Emițător:
FOR fiecare bit **s**[i]
IF baza ***b***[i] ≠ ***b'***[i]
elimină ***s***[i] din șirul ***s***
ENDIF
ENDFOR

Pasul 4 – Detectarea inamicului

Emițător și ***Receptor:***
FOR un subset de biți aleși aleatoriu din șirul ***s***
IF ***s'****[i]* ≠ ***s****[i]* și ***b'****[i]* = ***b****[i]*
Inamicul *a fost prezent*
ENDIF
ENDFOR

2.2.6 Diagrama funcțională a sistemului QKD BB84

Diagrama funcțională a sistemului de distribuire a cheilor cuantice BB84 este prezentat în figura 2.1.

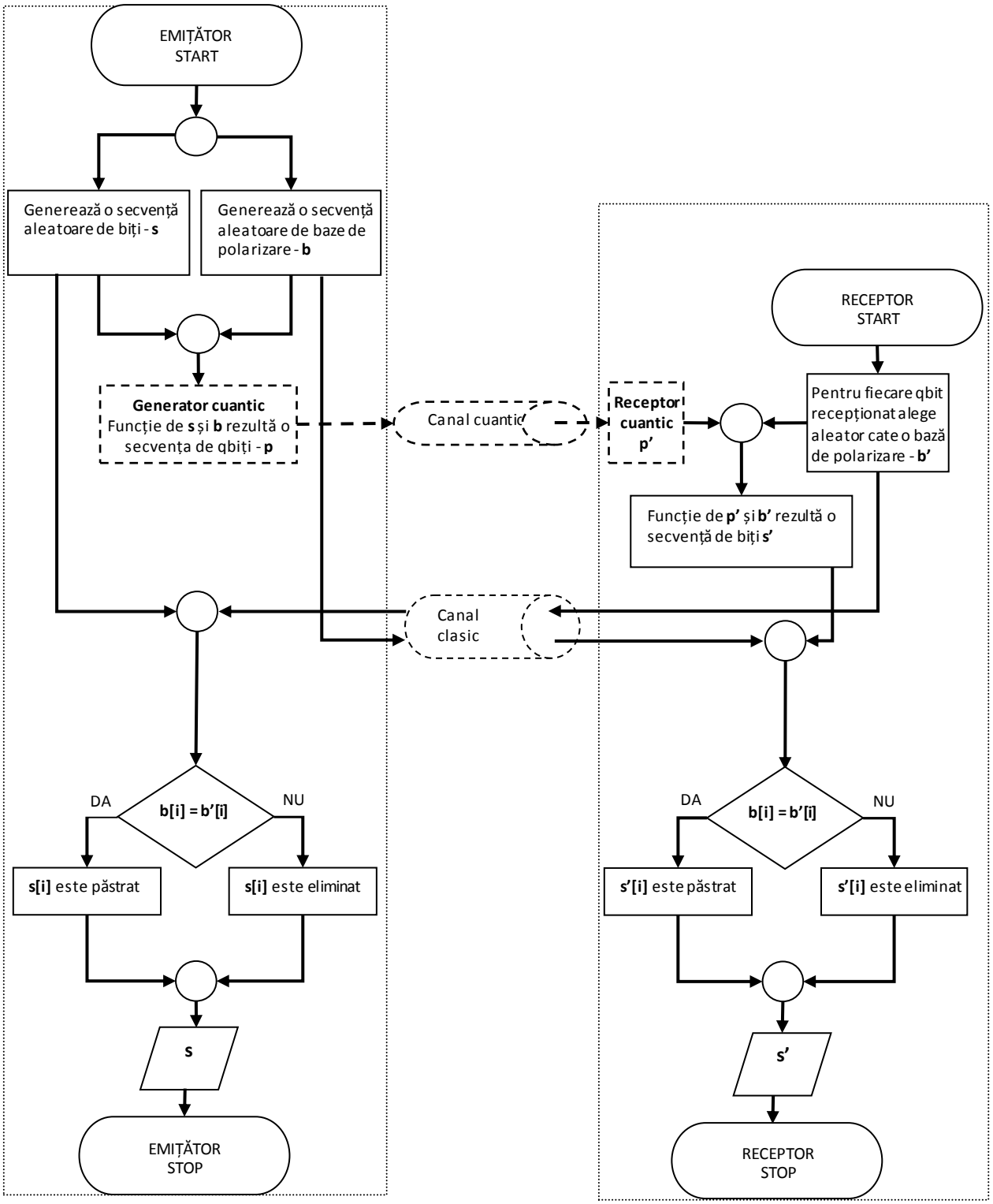

Figura 2.1. Diagrama sistemului QKD BB84

2.2.7 Exemplu teoretic

În continuare este prezentat un exemplu teoretic de distribuire a cheilor cuantice, între *emițător* și *receptor*, cu algoritmului BB84.

În tabelul 2.2 avem cazul ideal, în care *inamicul* nu este prezent iar transmisia se realizează fără pierderi de semnal pe canalul cuantic.

Bitul nr.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Secvența de biți - *emițător*	1	0	0	1	1	0	0
Bazele alese de *emițător*	L	L	D	D	D	L	D
Qbiții transmiși de emițător							
Bazele de polarizare - receptor							
Qbiții obținuți de receptor							
Bazele identice							
Qbiții păstrați							
Cheia finală	1	0	0				0

Tabelul 2.2. Algoritmul BB84 – cazul ideal

În tabelul 2.3 avem cazul în care *inamicul* este prezent iar transmisia se realizează fără pierderi de semnal.

Bitul nr.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Secvența de biți - *emițător*	1	0	0	1	1	0	0
Bazele alese de *emițător*	L	L	D	D	D	L	D
Qbiții transmiși de *emițător*							
Bazele aplicate de *inamic*							
Qbiții modificați de *inamic*							
Bazele de polarizare - *receptor*							
Qbiții obținuți de *receptor*							
Bazele identice							
Qbiții păstrați							
Cheia finală	1	0	**1**				0
Detectarea inamicului			**X**				

Tabelul 2.3. Algoritmul BB84 – inamicul este prezent

Inamicul este detectat că a intervenit pentru a citi qbitul numărul 3, datorită faptului că, deși *emițătorul* și *receptorul* au folosit pentru criptare respectiv decriptare baze diagonale, în cheia finală a *receptorului* bitul cu numărul 3 diferă de bitul numărul 3 din cheia *emițătorului*. Acest lucru se datorează datorită faptului că *inamicul*, a folosit pentru citirea qbitului numărul 3 o bază de polarizare orizontală care a schimbat direcția de polarizare a qbitului respectiv, din diagonală în orizontală.

2.3 Sistemul de distribuire a cheilor cuantice B92

Sistemul de distribuire a cheilor cuantice B92, propus de către Charles Bennet [9] în 1992, este similar cu sistemul BB84 cu diferența că *emițătorul* folosește pentru codificarea biților din cheia primară două stări non-ortogonale iar *receptorul* folosește pentru interpretarea qbiților toate cele patru stări de polarizare folosite în sistemul BB84.

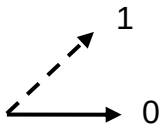

Figura 2.2. Stările de polarizare non-ortogonale ale *emițătorului*

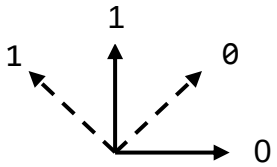

Figura 2.3. Stările de polarizare receptor

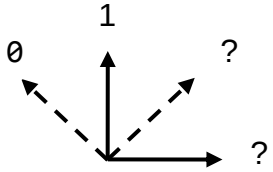

Figura 2.4. Interpretarea qbiților de către receptor

Conform legilor mecanicii cuantice, niciun tip de măsurare nu poate face distincție între doi fotoni polarizați în baze non-ortogonale, ca urmare este imposibil de identificat cu certitudine biții corespondenți. Pe de altă parte, orice încercare de a intercepta qbiții transmiși va alerta cele două părți care comunică.

Pentru implementarea sistemului de distribuire a cheilor cuantice B92, *emițătorul* și *receptorul* vor folosi pentru codificarea biților respectiv citirea fotonilor convenția stabilită în tabelul 2.4.

Emițător		
Stare	0^o	45^o
Qbit	→	↗
Bit	0	1

Receptor				
Stare	0^o	45^o	90^o	135^o
Qbit	→	↗	↑	↖
Rezultat	0	0	1	1
Bit	?	?	1	0

Tabelul 2.4. Stările de polarizare folosite de *emițător* și *receptor* în B92

2.3.1 Pașii sistemului QKD B92

1. *Emițătorul* generează o secvență aleatoare de biți - notată "***s***".
2. *Emițătorul* utilizează stările de polarizare (0^o, 45^o) pentru a reprezenta biții din "***s***". Notăm "***b***" secvența de polarizare.
3. *Emițătorul*, folosind un echipament special, creează o secvență "***p***" de fotoni polarizați, conform tabelului 2.4, a căror direcție de polarizare reprezintă biții din "***s***".
4. *Emițătorul* transmite qbiții din "***p***" prin fibră optică către *receptor*.
5. *Receptorul*, pentru fiecare qbit recepționat, alege aleator câte o bază de polarizare (liniar "L" sau diagonal "D"). Notăm cu "***b'***" secvența bazelor de polarizare aleasă.
6. *Receptorul*, măsoară fiecare qbit recepționat în parte, respectând bazele de polarizare alese la pct. 5 și îi comunică printr-un canal public *emițătorului* unde a primit un '0'.

   Dacă rezultatul va fi un '0', atunci ***s'*** = ? iar bitul respectiv va fi eliminat din ***s'***.

   Dacă rezultatul va fi un '1', atunci ***s'*** = 0 dacă ***b'*** = D și

   s' = 1 dacă ***b'*** = L
7. *Emițătorul* elimină din ***s*** biții unde a primit un '0' de la *receptor*.

2.3.2 Detectarea inamicului

Pentru detectarea *inamicului*, sistemul de distribuire a cheilor cuantice B92 folosește metoda QBER – Quantum Bit Error Rate care se definește ca fiind :

$$QBER = \frac{Q_I - Q_F}{Q_I} * 100, \tag{2.1}$$

unde Q_I reprezintă numărul de qbiți din șirul inițial, iar Q_F reprezintă numărul de qbiți din șirul final.

În absența *inamicului*, dacă pentru qbitul din poziția "i" vom avea ***s'***[i] ≠ ? atunci ***s'***[i] = ***s***[i].

Dacă inamicul interceptează ***s***[i] atunci rezultatul va fi că probabilitatea *Pr* {***s***[i] = ?} crește.

În concluzie *emițătorul* și *receptorul* vor detecta *inamicul*, datorită creșterii numărului de erori din cheia brută.

2.3.3 Secret key reconciliation

Etapa *Secret key reconciliation* [12] este o procedură de corectare a erorilor din cheia brută, care elimină :

- erorile generate de alegerea diferită a bazelor
- erorile generate de *Inamic*
- erorile generate de zgomote

Etapa *Secret key reconciliation* realizează o căutare binară, interactivă a erorilor. *Emițătorul* și *receptorul* împart secvența de biți rămasă (cheia brută) în blocuri de biți și vor compara paritatea fiecărui bloc. În cazul în care paritatea unui bloc de biți diferă, *emițătorul* și *receptorul* vor împărți blocul respectiv în blocuri mai mici și vor compara paritatea lor.

Acest proces va fi repetat până când bitul care diferă va fi descoperit și eliminat. Comunicațiile din această etapă se vor realiza pe un canal public nesecurizat la sfârșitul cărei se va obține cheia secretă.

2.3.4 Privacy amplification

Având în vedere faptul că, în etapa anterioară, comunicația s-a realizat pe un canal nesecurizat, există posibilitatea ca *inamicul* să dețină informații sensibile despre cheia secretă. Pentru a stabili o cheie perfect sigură, *emițătorului* și *receptorului* trebuie să mai realizeze o etapă : *Privacy amplification.* Această etapă constă într-o permutare a biților din cheia secretă și eliminarea unui subset de biți, care va fi realizată de către *emițător* și *receptor*.

La finalul acestei etape avem certitudinea că *emițătorul* și *receptorul*, dețin aceeași cheie unică, secretă care nu este cunoscută de *inamic*.

2.3.5 Pseudocodul sistemului QKD B92

Pentru sistemul de distribuire a cheilor cuantice B92, pseudocodul este următorul :

Pasul 1 – comunicarea pe canalul cuantic
Emițător:
generează aleatoriu dintre (0,1) șirul ***s***
FOR fiecare bit din **s**
generează un foton
IF **s**[i] = 0 THEN *polarizează* fotonul în starea (0^o) rezultă un qbit **p**[i] = →
IF **s**[i] = 1 THEN *polarizează* fotonul în starea (45^o) rezultă un qbit **p**[i] = ↗
trimite qbitul **p**[i] către ***Receptor***
ENDFOR

Receptor:
FOR fiecare qbit **p'**[i] recepționat
generează aleatoriu dintre ("L", "D") rezultând baza **b'**[i]
măsoară qbitul **p'**[i] în baza **b'**[i] rezultând o valoare **v** // unde $v \in \{0,1\}$
IF **v** = 0
s'[i] **= ?**
elimină bitul corespondent din ***s'***
trimite valoarea '0' către ***Emițător*** *// pe canalul clasic*
ELSE *// v = 1*
IF **b'**[i] = D
THEN ***s'***[i] = 0
ELSE *// b'[i] = R*
THEN ***s'***[i] = 1
ENDIF
ENDIF
ENDFOR

Pasul 2 – comunicarea pe canalul clasic
Emițător:
FOR fiecare valoare '0' recepționată de ***Receptor***
elimină bitul corespondent din ***s***
ENDFOR

Step 3 – Detectarea ***inamicului***
Emițător și Receptor:
IF Pr{***s'***[i] = ?} este mai mare decât normal (QBER) // Pr = probabilitatea
Inamicul *a fost prezent*
ENDIF

2.3.6 Diagrama funcțională a sistemului QKD B92

Diagrama funcțională a sistemului de distribuire a cheilor cuantice B92 este prezentat în figura 2.5.

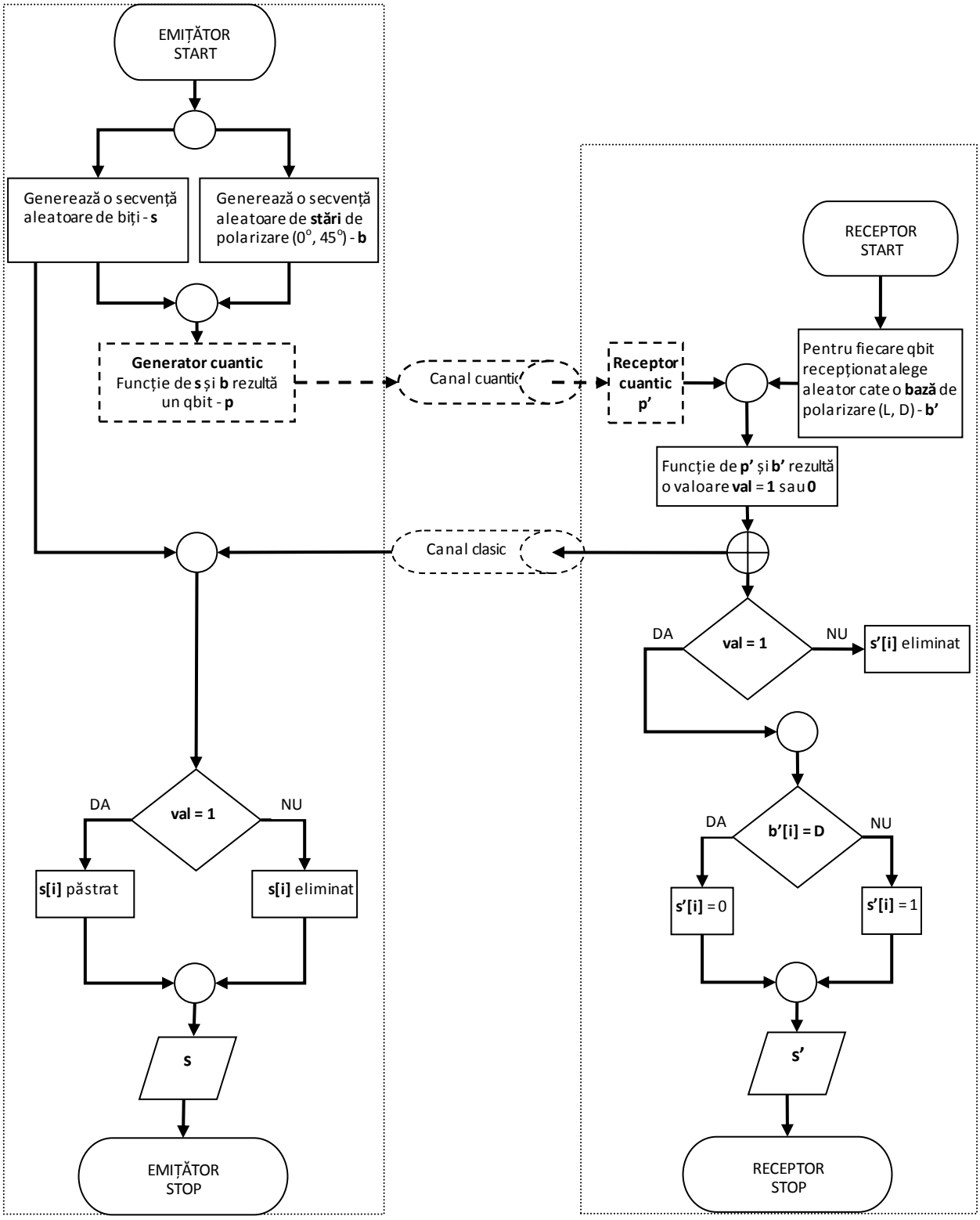

Figura 2.5. Diagrama sistemului QKD B92

2.4 Sistemul de distribuire a cheilor cuantice E91

Sistemul de distribuire a cheilor cuantice E91, cunoscut și ca protocolul Einstein-Podolsky-Rosen (EPR), numit după Artur Ekert care l-a publicat în 1991 [26], se bazează pe *entanglementul cuantic* al fotonilor și este o schemă de distribuție a cheilor cuantice de tip *entanglement*.

Fotonii *entanglați* sunt obținuți printr-un proces numit *spontaneous parametric down-conversion - SPDC* [76]. Astfel, un foton polarizat trece printr-un cristal nonliniar [46] și este divizat în doi fotoni *entanglați*.

În această schemă de distribuire a cheilor cuantice avem de a face cu trei părți implicate *Alice*, *Bob* și un *Generator cuantic. Alice* și *Bob* pot comunica printr-un canal clasic iar *Generatorul cuantic* va transmite, prin canalul cuantic, perechi entanglate de fotoni polarizați către cei doi. Astfel, cele două părți care comunică vor primi fiecare câte o particulă din perechea de qbiți entanglați.

Generatorul cuantic poate fi de sine stătător sau poate face parte din echipamentele lui *Alice* sau *Bob*.

Sunt utilizate 3 baze de polarizare non-ortogonale, $\left\{|0\rangle, |\frac{\pi}{2}\rangle\right\}$, $\left\{|\frac{\pi}{6}\rangle, |\frac{2\pi}{3}\rangle\right\}$ $\left\{|\frac{\pi}{3}\rangle, |\frac{5\pi}{6}\rangle\right\}$, iar pentru detectarea inamicului este folosită inegalitatea lui Bell [8].

Cele trei stări de polarizare entaglate sunt definite prin :

$$|\psi_A\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle_1 |\frac{\pi}{2}\rangle_2 - |\frac{\pi}{2}\rangle_1 |0\rangle_2\right) \quad (2.1)$$

$$|\psi_B\rangle = \frac{1}{\sqrt{2}}\left(|\frac{\pi}{6}\rangle_1 |\frac{2\pi}{3}\rangle_2 - |\frac{2\pi}{3}\rangle_1 |\frac{\pi}{6}\rangle_2\right) \quad (2.2)$$

$$|\psi_C\rangle = \frac{1}{\sqrt{2}}\left(|\frac{\pi}{3}\rangle_1 |\frac{5\pi}{6}\rangle_2 - |\frac{5\pi}{6}\rangle_1 |\frac{\pi}{3}\rangle_2\right) \quad (2.3)$$

Pentru fiecare bază de polarizare este stabilită următoarea convenție de polarizare a fotonilor :

Stare	$\|0\rangle$	$\|\frac{\pi}{2}\rangle$
Polarizare	0^o	90^0
Bit	0	1

Tabelul 2.5. Pentru $|\psi_A\rangle$

Stare	$\|\frac{\pi}{6}\rangle$	$\|\frac{2\pi}{3}\rangle$
Polarizare	30^o	120^0
Bit	0	1

Tabelul 2.6. Pentru $|\psi_B\rangle$

Stare	$\|\frac{\pi}{3}\rangle$	$\|\frac{5\pi}{6}\rangle$
Polarizare	60^o	150^0
Bit	0	1

Tabelul 2.7. Pentru $|\psi_C\rangle$

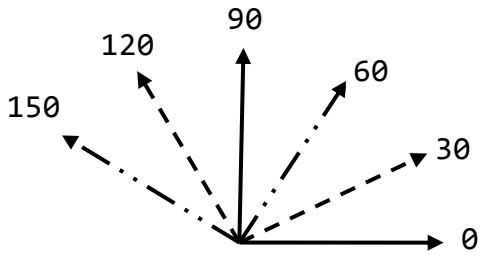

Figura 2.6. Stările de polarizare non-ortogonale folosite în algoritmul E91

Astfel, conform entanglementului cuantic descoperit de către Einstein-Podolsky-Rosen, dacă un foton este citit în starea de polarizare $|0\rangle$ celălalt foton din perechea entanglată va avea starea de polarizare $|\frac{\pi}{2}\rangle$ și viceversa.

Dacă fotonul este citit în starea de polarizare $|\frac{\pi}{6}\rangle$ celălalt foton din perechea entanglată va avea starea de polarizare $|\frac{2\pi}{3}\rangle$ și viceversa.

Dacă fotonul este citit în starea de polarizare $|\frac{\pi}{3}\rangle$ celălalt foton din perechea entanglată va avea starea de polarizare $|\frac{5\pi}{6}\rangle$ și viceversa.

Ca și în cazul celorlalți algoritmi de distribuire a cheilor cuantice există două etape de comunicare, una printr-un canal cuantic (fibră optică) și una printr-un canal public.

2.4.1 Pașii sistemului QKD E91

1. *Generatorul,* produce perechi de qbiți entanglați, polarizați aleatoriu în una din stările bazelor $\{|\psi_A\rangle, |\psi_B\rangle, |\psi_C\rangle\}$, pe care îi transmite unul către *Alice* și celălalt către *Bob*.
2. *Alice*, pentru fiecare qbit primit, alege aleator dintre bazele de polarizare $\left\{|0\rangle, |\frac{\pi}{2}\rangle\right\}$, $\left\{|\frac{\pi}{6}\rangle, |\frac{2\pi}{3}\rangle\right\}$ sau $\left\{|\frac{\pi}{3}\rangle, |\frac{5\pi}{6}\rangle\right\}$, cu care măsoară qbiții primiți.
3. *Bob*, pentru fiecare qbit primit, alege aleator dintre bazele de polarizare $\left\{|0\rangle, |\frac{\pi}{2}\rangle\right\}$, $\left\{|\frac{\pi}{6}\rangle, |\frac{2\pi}{3}\rangle\right\}$ sau $\left\{|\frac{\pi}{3}\rangle, |\frac{5\pi}{6}\rangle\right\}$, cu care măsoară qbiții primiți.
4. *Alice* și *Bob* comunică printr-un canal public unde au ales aceleași baze de polarizare. Biții pentru care nu au fost alese aceleași baze de polarizare sunt eliminați din cheia brută dar sunt folosiți pentru detectarea unui eventual *inamic*.

2.4.2 Detectarea Inamicului

Spre diferență de alți algoritmi, algoritmul E91, în loc să elimine complet biții eliminați, îi utilizează pentru a detecta prezența sau absența *inamicului*. *Alice* și *Bob* comunică printr-un canal public și își compară bazele și biții eliminați pentru a determina dacă inegalitatea lui Bell este satisfăcută sau nu.

Inamicului este detectat prin analizarea inegalității lui Bell, dacă este respectată inegalitatea lui Bell [8] rezultă că *inamicul* a fost prezent.

Inegalitatea lui Bell transpusă în E91:

$$1 + P(B,C) \geq |P(A,B) - P(A,C)| \tag{2.4}$$

unde $P(i,j)$ reprezintă valoarea medie a rezultatelor măsurării a primei particule în baza i iar celei de a doua în baza j, dintre cele trei baze posibile $A = \left\{|0\rangle, |\frac{\pi}{2}\rangle\right\}$, $B = \left\{|\frac{\pi}{6}\rangle, |\frac{2\pi}{3}\rangle\right\}$, $C = \left\{|\frac{\pi}{3}\rangle, |\frac{5\pi}{6}\rangle\right\}$.

Definim $P(\neq |i,j)$ ca fiind probabilitatea ca cei doi qbiți entanglați, dintre cei eliminați, să nu se potrivească, datorită faptului că *Alice* a ales baza de polarizare i și *Bob* a ales baza de polarizare j [48].

Definim :

$$P(= |i,j) = \ 1 - P(\neq |i,j). \tag{2.5}$$

Definim :

$$\Delta(i,j) = P(\neq |i,j) - P(= |i,j) \tag{2.6}$$

Definim :

$$\beta = 1 + \Delta(\mathrm{B},\mathrm{C}) - |\Delta(\mathrm{A},\mathrm{B}) - \Delta(\mathrm{A},\mathrm{C})| \tag{2.7}$$

În acest caz, inegalitatea lui Bell se va reduce la :

$$\beta \geq 0 \tag{2.8}$$

2.4.3 Secret key reconciliation

Etapa *secret key reconciliation* [12] este o procedură de corectare a erorilor din cheia brută, care elimină :

- erorile generate de alegerea diferită a bazelor
- erorile generate de *Inamic*
- erorile generate de zgomote

Etapa *Secret key reconciliation* realizează o căutare binară, interactivă a erorilor. *Emițătorul* și *receptorul* împart secvența de biți rămasă (cheia brută) în blocuri de biți și vor compara paritatea fiecărui bloc. În cazul în care paritatea unui bloc de biți diferă, *emițătorul* și *receptorul* vor împărți blocul respectiv în blocuri mai mici și vor compara paritatea lor.

Acest proces va fi repetat până când bitul care diferă va fi descoperit și eliminat. Comunicațiile din această etapă se vor realiza pe un canal public nesecurizat la sfârșitul cărei se va obține cheia secretă.

2.4.4 Privacy amplification

Având în vedere faptul că, în etapa anterioară, comunicația s-a realizat pe un canal nesecurizat, există posibilitatea ca *inamicul* să dețină informații sensibile despre cheia secretă. Pentru a stabili o cheie perfect sigură, *emițătorului* și *receptorului* trebuie să mai realizeze o etapă : *Privacy amplification.* Această etapă constă într-o permutare a biților din cheia secretă și eliminarea unui subset de biți, care va fi realizată de către *emițător* și *receptor*.

La finalul acestei etape avem certitudinea că *emițătorul* și *receptorul*, dețin aceeași cheie unică, secretă care nu este cunoscută de *inamic*.

2.4.5 Pseudocodul sistemului QKD E91

Pentru sistemul de distribuire a cheilor cuantice E91, pseudocodul este următorul :

Pasul 1 – comunicarea pe canalul cuantic

Generator:

generează fotoni polarizați aleatoriu în una din bazele $\left\{|0\rangle, |\frac{\pi}{2}\rangle\right\}$, $\left\{|\frac{\pi}{6}\rangle, |\frac{2\pi}{3}\rangle\right\}$, $\left\{|\frac{\pi}{3}\rangle, |\frac{5\pi}{6}\rangle\right\}$

FOR fiecare foton polarizat

prin SPDC fotonul polarizat este *divizat* în doi qbiți entanglați ***p***[i] și ***p'***[i]

trimite către ***Alice*** qbitul ***p***[i] iar către ***Bob*** qbitul ***p'***[i]

ENDFOR

Pasul 2 – comunicarea pe canalul clasic

Alice:

FOR fiecare qbit ***p***[i] recepționat

alege aleatoriu ***b***[i] dintre bazele de polarizare $\left\{|0\rangle, |\frac{\pi}{2}\rangle\right\}, \left\{|\frac{\pi}{6}\rangle, |\frac{2\pi}{3}\rangle\right\}, \left\{|\frac{\pi}{3}\rangle, |\frac{5\pi}{6}\rangle\right\}$

măsoară qbitul ***p***[i] în baza ***b***[i] rezultând bitul ***s***[i]

transmite baza ***b***[i] către ***Bob***

recepționează baza ***b'***[i] de la ***Bob***

IF baza ***b***[i] ≠ ***b'***[i]

elimină ***s***[i] din șirul ***s***

pune ***s***[i] din șirul ***v***

ENDIF

ENDFOR

Bob:

FOR fiecare qbit ***p'***[i] recepționat

alege aleatoriu ***b'***[i] dintre bazele de polarizare $\left\{|0\rangle, |\frac{\pi}{2}\rangle\right\}, \left\{|\frac{\pi}{6}\rangle, |\frac{2\pi}{3}\rangle\right\}, \left\{|\frac{\pi}{3}\rangle, |\frac{5\pi}{6}\rangle\right\}$

măsoară qbitul ***p'***[i] în baza ***b'***[i] rezultând bitul ***s'***[i]

transmite baza ***b'***[i] către ***Alice***

recepționează baza ***b***[i] de la ***Alice***

IF baza ***b'***[i] ≠ ***b***[i]

elimină ***s'***[i] din șirul ***s'***

pune ***s'***[i] din șirul ***v'***

ENDIF

ENDFOR

Pasul 3 – Detectarea inamicului

Emițător și ***Receptor:***

FOR biții din ***v*** și ***v'***

testează inegalitatea lui Bell

IF *inegalitatea lui Bell* ≥ 0

Inamicul a fost prezent

ELSE

Inamicul nu a fost prezent

ENDIF

ENDFOR

2.4.6 Diagrama funcțională a sistemului QKD E91

Diagrama funcțională a sistemului de distribuire a cheilor cuantice B92 este prezentat în figura 2.7.

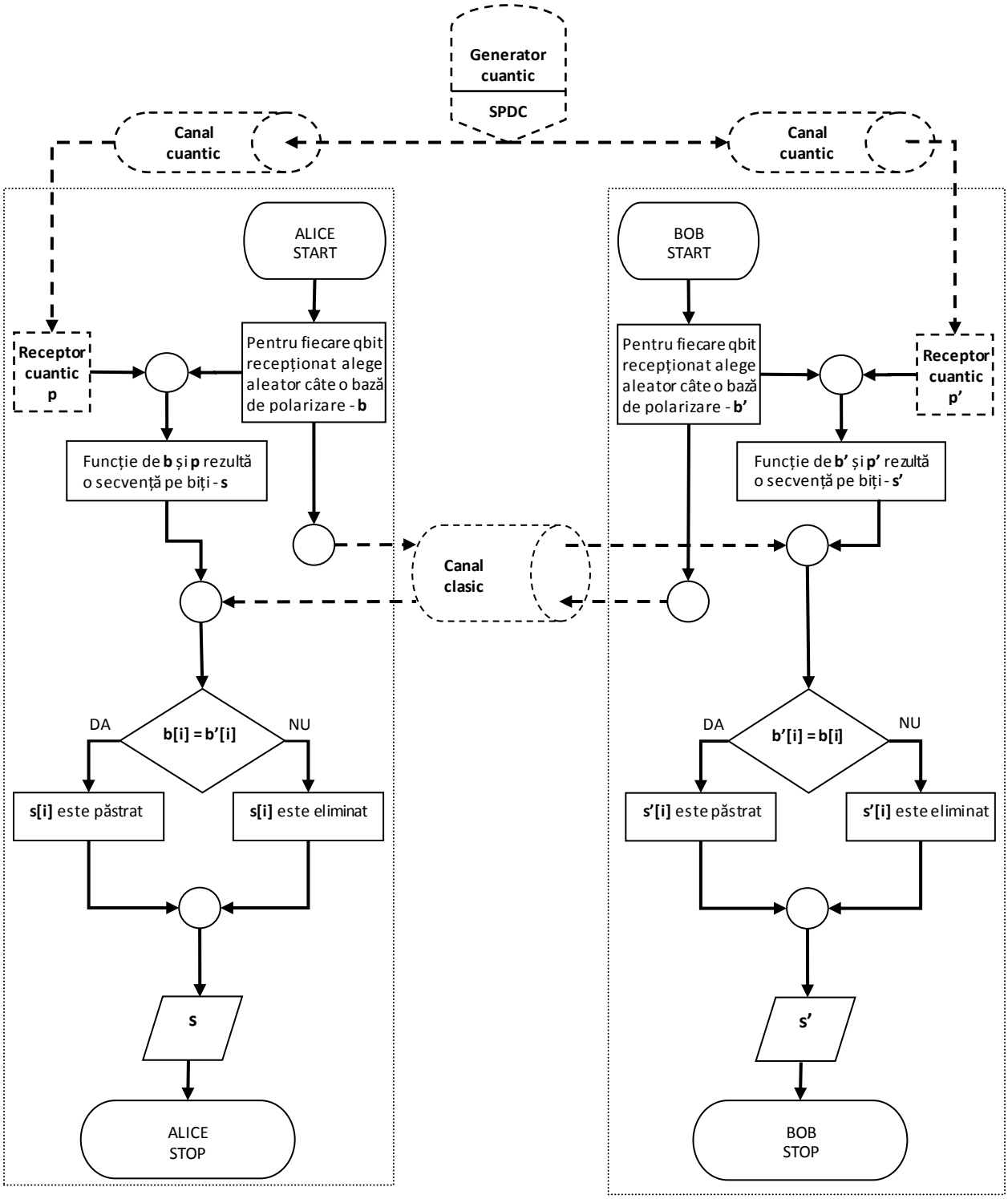

Figura 2.7. Diagrama sistemului QKD E91

2.5 Concluzii

Schemele de distribuire a cheilor cuantice se împart în scheme de tip single photon, care folosesc un singur foton polarizat și de tip entanglement care folosesc o pereche de fotoni entanglați.

În acest capitol am prezentat cele mai cunoscute și utilizate scheme de distribuire a cheilor cuantice care vor fi folosite ulterior pentru dezvoltarea contribuțiilor proprii din cadrul tezei.

Cheia finală, obținută cu ajutorul oricărei scheme de distribuire a cheilor cuantice poate fi utilizată împreună cu algoritmul one-time pad [71,72] pentru a crea un criptosistem perfect sigur.

Se observă că toate sistemele de distribuire a cheilor cuantice efectuează, la sfârșitul transmisiei cuantice după etapa de comunicare a bazelor de polarizare pe canalul clasic, etapele de verificare și securizare a transmisiei cum ar fi *detectarea inamicului*, *Secret key reconciliation* și *Privacy amplification*, în urma cărora rezultă cheia finală. Acest lucru poate da naștere mai multor inconveniente și vulnerabilități legate de securitatea transmisiei și a cheii finale :

(1) cheia finală poate avea dimensiuni mai mici decât dimensiunea necesară
(2) transmisia ar trebui reluată
(3) inamicul este detectat după transmiterea tuturor qbiților din cheie.

Detectarea inamicului la sfârșitul transmisiei cuantice, după comunicarea bazelor de polarizare pe canalul clasic, crează posibilități de atac asupra acestor schemelor de distribuire a cheilor cuantice

În cadrul acestei teze vor fi analizate toate aceste vulnerabilități și vor fi propuse soluții de contracarare a acestora în capitolele 4 și 6.

3. Analiza securității sistemelor de distribuire a cheilor cuantice

3.1 Introducere

Istoria criptografiei este plină de exemple de algoritmi criptografici care se credeau că sunt siguri la momentul descoperirii lor dar care s-au dovedit, după mai mulți ani, a fi vulnerabili la noi tipuri de atacuri care inițial nu erau cunoscute. Având în vedere acest fapt, trebuie să ne așteptăm ca și sistemele de distribuire a cheilor cuantice, considerate a fi de securitate necondiționată, vor avea vulnerabilități care nu pot fi anticipate în momentul proiectării lor, mai ales că o mare varietate de tipuri de atac [29,45,47,49-54,57,70,74,75,78], unele pur teoretice altele practice, au fost propuse pentru sistemele actuale de distribuire a cheilor cuantice

Securitatea necondiționată a sistemelor de distribuire a cheilor cuantice – QKD (Quantum Key Distribution) [12,30,37,43,48,55,61,66,68], a fost demonstrată doar pentru modelele matematice ale schemelor de distribuire a cheilor cuantice. În practică, această securitate necondiționată nu poate fi atinsă datorită imperfecțiunilor tehnice ale dispozitivelor utilizate pentru polarizarea respectiv citirea polarizării fotonilor, implicate în schimbul de chei cuantice. Având în vedere că aceste dispozitive se îmbunătățesc și vor fi îmbunătățite odată cu noile descoperiri științifice, dezvoltarea sistemelor de distribuire a cheilor cuantice se realizează pentru cazul ideal. Cu alte cuvinte tendința este de a crea noi modele matematice ale sistemelor de distribuire a cheilor cuantice, de a îmbunătăți și dezvolta sistemele existente rezultând astfel o abordare mai mult teoretică a sistemelor QKD.

Acest capitol prezintă principalele tipuri de atac și principalele metode de detectare a atacurilor asupra sistemelor de distribuire a cheilor cuantice.

3.2 Vulnerabilitățile sistemelor QKD

Vulnerabilitățile teoretice și cele practice ale sistemelor de distribuire a cheilor cuantice – QKD au constituit dintotdeauna principalul punct de plecare al metodelor de atac asupra acestor sisteme.

Un aspect foarte important de care trebuie să ținem cont este că sistemele de distribuire a cheilor cuantice sunt din punct de vedere teoretic perfect sigure dar echipamentele cuantice pentru transmisie și mai ales pentru recepție sunt departe de a fi perfecte. Aceste imperfecțiuni tehnice ale dispozitivelor cuantice deschid calea unor noi metode de atac, unele teoretice altele practice, asupra sistemelor de distribuire a cheilor cuantice.

Deși tehnologia actuală nu permite implementarea practică a tuturor tipurilor de atacuri, este foarte important de luat în considerare toate formele posibile de atac asupra sistemelor de distribuire a cheilor cuantice și de a realiza metode, chiar și teoretice, de a le contracara, pentru a asigura securitatea necondiționată prezentă și viitoare a acestor sisteme.

Există mai multe tipuri de atacuri asupra sistemelor de distribuire a cheilor cuantice, iar dintre cele mai importante amintim :

(1) Atacul de tip Intercept/Resend (Faked-State)
(2) Atacul de tip Photon Number Splitting (PNS)
(3) Atacul de tip Man-in-the-Middle (MiM)
(4) Atacul de tip Quantum Cloning

3.2.1 Atacul de tip Intercept-Resend

Atacul de tip Intercept-Resend [52] numit și Faked-State, este cel mai întâlnit tip de atac folosit asupra sistemelor de distribuire a cheilor cuantice. *Inamicul*, întrerupe canalul cuantic, măsoară fiecare qbit recepționat de la *emițător* în una dintre cele două baze de polarizare și transmite către *receptor* alți qbiți polarizați în baze de polarizare corespunzătoare cu rezultatele obținute de către el, fără a lăsa urme ale atacului [28].

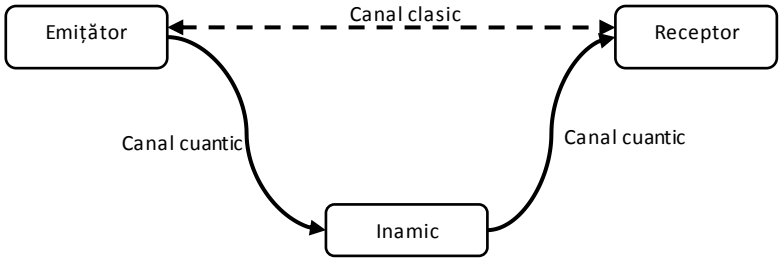

Figura 3.1. Atac de tip Intercept-Resend

Folosirea a două baze de polarizare - cazul sistemelor de distribuire a cheilor cuantice BB84 și B92 - îi dă *inamicului* șansa de a obține 50% dintre măsurători compatibile cu qbiții transmiși de către *emițător*, care vor da rezultate corecte iar restul de 50% vor da rezultate aleatoare. Astfel *inamicul* și *receptorul* au șanse egale de a măsura corect qbiții transmiși de către *emițător*. Atacul este nedetectabil prin tehnicile actuale pentru dispozitive la care rata erorilor este mai mare de 25% [31].

3.2.2 Atacul de tip Photon Number Splitting

Atacul de tip PNS [17,35] este cel mai bun tipul de atac asupra unui sistem de distribuire a cheilor cuantice atunci când pentru generarea fotonilor se folosește un laser cu impuls atenuat.

Acest tip de atac se bazează pe faptul că, datorită imperfecțiunilor echipamentelor cuantice, pentru codificarea unui bit sunt utilizați unul sau mai mulți fotoni polarizați. Altfel spus pentru codificarea unui bit, în loc să plece de la *emițător* un impuls care conține un singur foton polarizat, pleacă un impuls care este format din mai mulți fotoni polarizați identic.

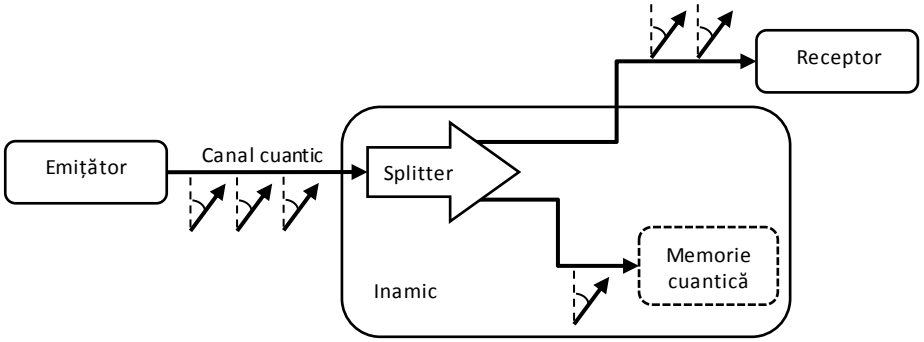

Figura 3.2. Atac de tip PNS

Inamicul, verifică numărul de fotoni din fiecare impuls generat de către *emițător*, fără a perturba polarizarea fotonilor și blochează toate impulsurile care conțin doar un singur foton. Pentru impulsurile care conțin mai mulți fotoni extrage un singur foton pe care îl memorează într-o memorie cuantică lăsând restul să plece spre *receptor*. Ulterior, când *emițătorul* și *receptorul* realizează compararea bazelor de polarizare folosite, *inamicul* măsoară fotonii din memoria cuantică în conformitate cu bazele anunțate, obținând astfel cheia secretă.

Datorită faptului că fotonii transmiși de către *emițător* nu sunt alterați de către *inamic*, acest tip de atac nu poate fi detectat de către *receptor* sau *emițător* prin tehnicile actuale.

3.2.3 Atacul de tip Man-in-the-Middle

Sistemele de distribuire a cheilor cuantice sunt vulnerabile la atacurile de tip Man-in-the-Middle atunci când nu sunt folosite metode de autentificare între emițător și receptor deoarece inamicul poate pretinde că este receptorul respectiv emițătorul.

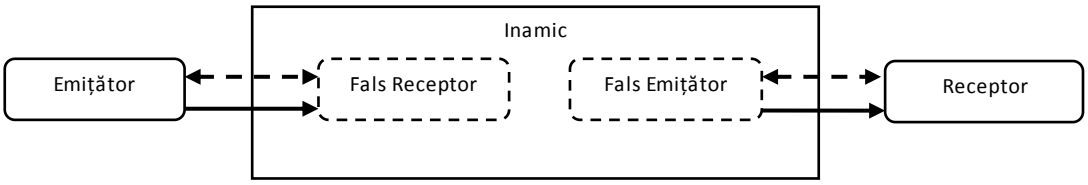

Figura 3.3. Atac de tip Man-in-the-Middle

Inamicul întrerupe ambele canale de comunicații, cuantic și clasic, dintre *emițător* și *receptor*, pretinzând pentru *emițător* că este *receptorul* iar pentru *receptor* că este *emițătorul*. Au fost propuse mai multe metode de implementare a acestui tip de atac folosind o a trei-a parte [79] sau teoria haosului [34]. Acest tip de atac nu poate fi detectat deoarece qbiții transmiși de către *emițător* nu ajung la *receptor*.

3.2.4 Atacul de tip Quantum Cloning

Atacul de tip quantum cloning este o metodă pur teoretică de atac care folosește un dispozitiv de clonare propus de către N. Gisin și B. Huttner [27] numit "Pretty Good Quantum Copying Machine" (PGQCM) și o memorie cuantică.

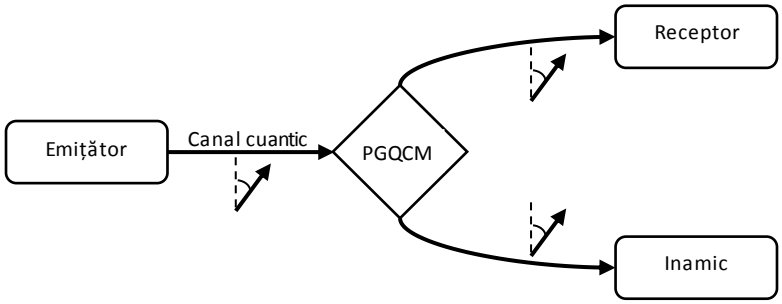

Figura 3.4. Atac de tip Quantum Cloning

Inamicul interceptează fiecare foton transmis de către *emițător* și folosind o mașină de clonare obține doi fotoni identici. Unul dintre fotoni este transmis către *receptor* iar celălalt este păstrat de către *inamic* într-o memorie cuantică. În timpul etapei *bases reconciliation, inamicul* scoate cate un foton din memoria cuantică și îl măsoară în funcție de baza de polarizare anunțată.

Datorită faptului că fotonii transmiși de către *emițător* nu sunt alterați de către *inamic*, acest tip de atac nu poate fi detectat de către *receptor* sau *emițător* prin metodele actuale de detectare a inamicului.

3.3 Detectarea atacurilor asupra QKD

Securitatea sistemelor de distribuire a cheilor cuantice este dată și de eficacitatea metodei de detectare a unor eventuale atacuri.

Există mai multe metode de detectare a atacurilor asupra sistemelor de distribuire a cheilor cuantice.

(1) Metoda clasică – identificarea qbiților alterați de către *inamic*
(2) QBER – estimarea ratei de erori din cheia finală
(3) Inegalitatea lui Bell

3.3.1 Metoda clasică

Detectarea inamicului se realizează la finalul transmisiei cuantice, după comunicarea bazelor de polarizare pe canalul public la sfârșitul etapei bases reconciliation. Emițătorul și receptorul, compară o parte din biții din cheia primară, pentru care au ales baze de polarizare identice, iar dacă aceștia diferă rezultă că inamicul a intervenit pentru a citi qbiții respectivi, modificându-le polarizarea.

3.3.2 Quantum Bit Error Rate – QBER

Această metodă constă în calcularea procentului de erori din cheia finală, obținută la finalul transmisiei cuantice, după etapa de comunicare a bazelor de polarizare pe canalul public [69].

Quantum bit error rate se definește ca fiind :

$$QBER = \frac{Q_E}{Q_T} * 100, \quad (3.1)$$

unde Q_E reprezintă numărul de qbiți care au generat erori, iar Q_T reprezintă numărul total de qbiți

Metoda QBER – Quantum Bit Error Rate pentru detectarea *inamicului* se poate aplica majorității sistemelor de distribuire a cheilor, fiecare sistem având propria rată de erori acceptată, orice depășire a ei însemnând intervenția unui *inamic*.

3.3.3 Inegalitatea lui Bell

Inegalitatea lui Bell [8] poate fi folosită pentru a detecta prezența sau absența unui *inamic*. Astfel, dacă inegalitatea lui Bell este satisfăcută rezultă că *inamicul* a intervenit pentru a citi qbiții transmiși de către *emițător* iar în caz contrar *inamicul* nu a fost prezent.

3.4 Concluzii

Securitatea sistemelor de distribuire a cheilor cuantice este una dintre principalele priorități și direcții de cercetare în criptografia cuantică. Această securitate are în vedere atât vulnerabilitatea sistemelor cuantice la atacurile unui eventual inamic cât și metodele de detectare a atacurilor asupra sistemelor de distribuire a cheilor cuantice.

Chiar dacă multe metode de atac sunt pur teoretice, acestea trebuiesc luate în considerație și găsite soluții de detectare a lor, datorită faptului că domeniul criptografiei cuantice este în plină dezvoltare.

Observăm că anumite tipuri de atacuri nu pot fi depistate cu metodele de detectare actuale, ceea ce reprezintă o mare breșă de securitate a sistemelor de distribuire a cheilor cuantice.

În cadrul acestei teze vor fi analizate toate aceste vulnerabilități și vor fi propuse soluții de contracarare a acestora în capitolul 5.

4. Proiectarea protocolului Base Selection and Transmission Synchronization

4.1 Introducere

Algoritmii criptografici cuantici existenți realizează un schimb de chei, între un emițător și un receptor, rezultând la finalul transmisiei cuantice o cheie finală unică și secretă care va fi utilizată în coroborare cu algoritmul one-time pad [71,72], pentru a realiza o comunicare perfect sigură. În cazul acestor algoritmi transmisia cheii se realizează pe canalul cuantic iar comunicarea, cu algoritmul one-time pad, se realizează pe un canal clasic.

În acest capitol am propus un protocol de comunicații bidirecțional cuantic numit Base Selection and Transmission Synchronization – BSTS [2,3] în care comunicarea se va realiza exclusiv pe un canal cuantic.

Acest protocol criptografic cuantic de comunicații, folosește biții din *cheia finală*, obținută după etapele *Secret key reconciliation* și *Privacy amplification*, ale oricărui algoritm de distribuire a cheilor cuantice și stabilește cu exactitate parametrii utilizați, de către *emițător* și *receptor*, pentru realizarea transmisiei cuantice.

Sunt prezentate etapele, pseudocodul și diagrama funcțională a noului protocol Base Selection and Transmission Synchronization – BSTS.

4.2 Necesitatea protocolului BSTS

După etapele *Secret key reconciliation* și *Privacy Amplification*, ***cheia finală***, unică, secretă și sigură, obținută de către ***emițător*** și *receptor* este foarte mică comparativ cu secvența inițială pregătită de către *emițător* și de cele mai multe ori insuficient de mare pentru a fi utilizată pentru o criptare cu algoritmul one-time pad [71,72].

În figura 4.1 este prezentată, diferența dintre dimensiunea cheii inițiale și dimensiunea cheii finale, la sfârșitul transmisiei cuantice, după etapele *Secret key reconciliation* și *Privacy Amplification*.

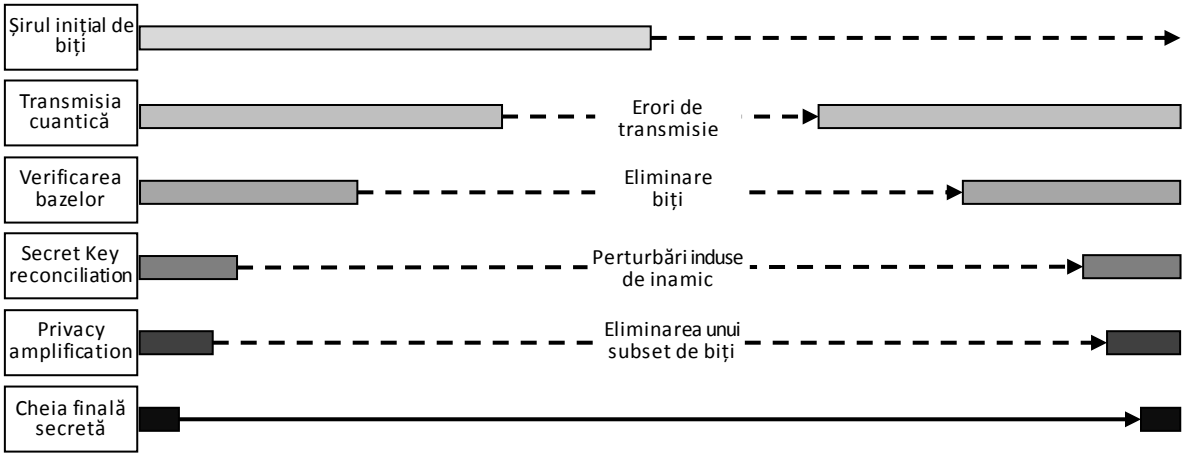

Figura 4.1. Diferența dintre cheia inițială și cea finală

Securitatea necondiționată a algoritmului one-time pad [71,72] fost demonstrată de către Claude Shannon [62], cu condițiile ca, cheia de criptare să fie secretă, să nu fie folosită decât o singură dată și să fie de aceeași lungime cu textul clar.

Cu toate că în urma schimbului de chei cuantice avem garanția că am obținut o cheie finală secretă și nu o vom folosi decât o singură dată, nu avem garanția că vom obține de fiecare dată o cheie finală de criptare, de dimensiunea textului clar pe care dorim să-l transmitem, iar acest fapt constituie un inconvenient major al sistemelor de distribuire a cheilor cuantice existente.

Având în vedere aceste aspecte propun un protocol de comunicare cuantic, bidirecțional, între un emițător și un receptor, pe care l-am numit protocolul Base Selection and Transmission Synchronization – BSTS [2,3].

4.3 Premisele protocolului BSTS

Parametrii vizați în acest protocol sunt:

(1) ce pereche de baze de polarizare dintre cele trei, L (liniară), D (diagonală) și C (circulară), va fi folosită;
(2) intervalul de timp la care *emițătorul* va emite un qbit iar *receptorul* îl va citi;
(3) ce bază de polarizare va fi folosită pentru codificarea fiecărui bit în parte;
(4) verificarea biților recepționați la finalul transmisiei cuantice și retransmiterea blocurilor care conțin erori.

Pentru codificarea biților vom stabili următoarea convenție de codificare și reprezentare a qbiților :

Baza	**L (Liniar)**	**D (Diagonal)**	**C (Circular)**	**C (Circular)**	**L (Liniar)**	**D (Diagonal)**
Polarizare	0^o	45^o	spinL	spinR	90^o	135^o
Qbit						
Bit	0	0	0	1	1	1

Tabelul 4.1. Polarizarea fotonilor - qbiți

Prin acest protocol cuantic de comunicații *emițătorul* și *receptorul* vor stabili în comun, în funcție de biții din *cheia finală*, care pereche, dintre cele trei baze de polarizare, *liniar-diagonal*, *liniar-circular* sau *diagonal-circular* vor fi folosite pentru polarizarea fotonilor – acest proces se va numi *base selection*.

Tot în funcție de biții din *cheia finală*, protocolul cuantic de comunicații, va stabili intervalul de timp în care *emițătorul* va transmite un foton real iar *receptorul* va citi fotonul respectiv, în restul timpului *emițătorul* va transmite fotoni falși pentru inducerea în eroare a unui eventual *inamic* – acest proces se va numi *transmission synchronization*.

În final, în funcție de biții rămași din *cheia finală*, *emițătorul* și *receptorul*, vor ști exact ce baze de polarizare să aplice pentru fiecare foton pe care urmează să-l transmită respectiv să-l recepționeze.

După terminarea transmisiei, *emițătorul* și *receptorul*, împart secvența de biți obținută în blocuri de biți și vor compara paritatea fiecărui bloc. În cazul în care paritatea unui bloc de biți diferă, se va relua transmisia blocului respectiv.

Schema bloc a unui sistem cuantic de comunicații bazat pe protocolul Base Selection and Transmission Synchronization este prezentată în figura 4.2.

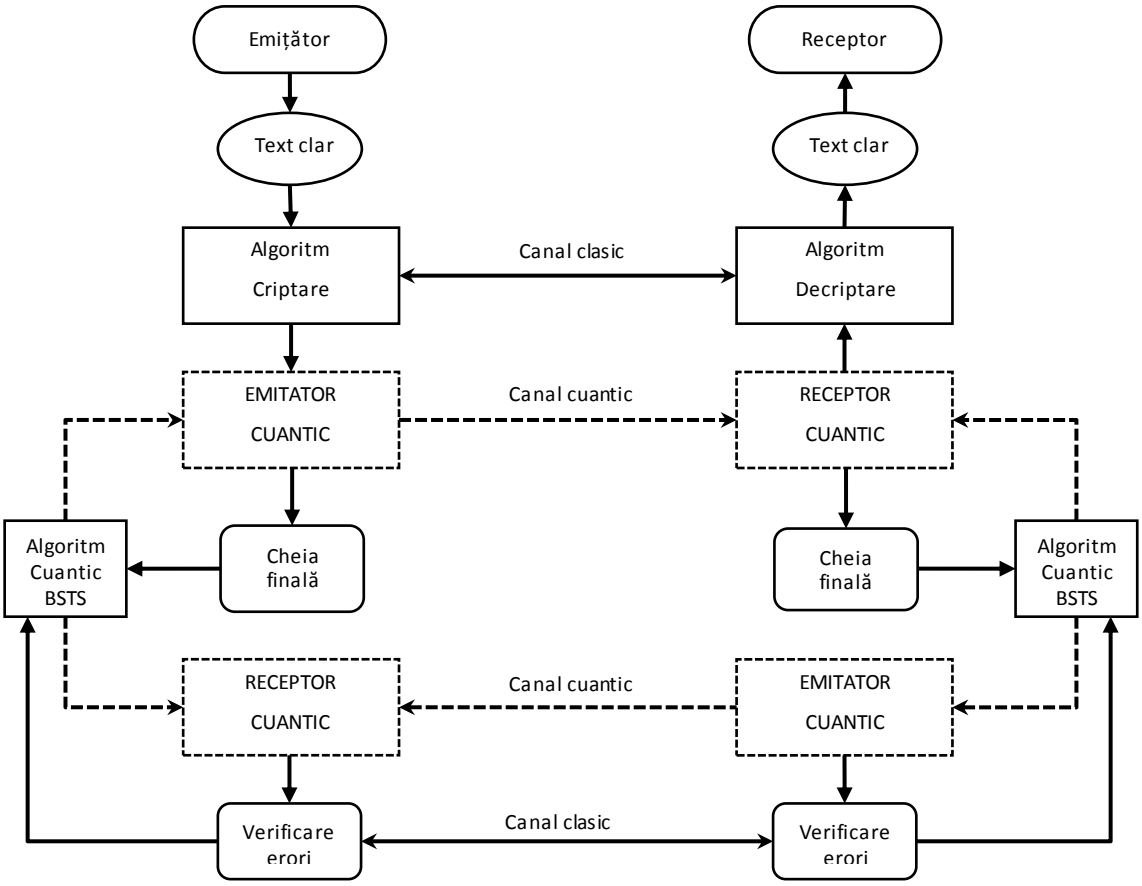

Figura 4.2. Schema bloc a algoritmului de comunicații BSTS

4.4 Pașii protocolului BSTS

Algoritmul va fi executat simultan de către *emițător* și *receptor* și are următoarele etape :

1. Primii doi biți din *cheia finală* vor fi utilizați pentru a stabili bazele de polarizare, care vor fi folosite pentru a polariza fotonii ce urmează a fi transmiși, conform tabelului :

Bitul 1	Bitul 2	Baza 1	Baza 2
0	0	L	D
0	1	L	C
1	0	C	D
1	1	L	D

Tabelul 4.2. Selectarea bazelor în funcție de biții 1 și 2 din *cheia finală – base selection*

2. Următorii patru biți din *cheia finală* vor forma un număr în binar a cărui conversie în zecimal, +1, va reprezenta intervalul de timp, exprimat în microsecunde, la care se va realiza transmiterea respectiv recepționarea unui qbit real. *Emițătorul* va transmite qbiți reali către *receptor* la intervalele de timp stabilite în această etapă, în restul timpului *emițătorul* va transmite fotoni falși polarizați aleator.

Bitul 3	**Bitul 4**	**Bitul 5**	**Bitul 6**	**Număr în baza 10**	**Timp (μs) (Număr +1)**
0	0	0	0	0	1
0	0	0	1	1	2
0	0	1	0	2	3
0	0	1	1	3	4
.....				.....	
1	1	0	1	13	14
1	1	1	0	14	15
1	1	1	1	15	16

Tabelul 4.3. Stabilirea intervalului de timp al transmisiei - *transmission synchronization*

3. Biții rămași din *cheia finală* vor reprezenta bazele de polarizare pentru fiecare bit care va fi transmis, respectiv recepționat, conform tabelului 4.4 :

Dacă Bitul este	0	1
Baza de polarizare	Baza 1	Baza 2

Tabelul 4.4. Stabilirea bazelor pentru polarizarea fiecărui bit

4. După ultimul bit din *cheia finală*, procesul de polarizare al fotonilor va continua de la bitul numărul 7 al cheii finale până la terminarea mesajului care se dorește a fi transmis.
5. *Emițătorul și receptorul* comunică pe canalul clasic și împart secvența de biți obținută în blocuri de biți comparând paritatea fiecărui bloc. În cazul în care paritatea unui bloc de biți diferă, se va relua transmisia blocului respectiv.

4.5 Exemplu teoretic

Presupunem că, *cheia finală* obținută după etapele *Secret key reconciliation* și *Privacy amplification*, a oricărei scheme de distribuire a cheilor cuantice, este formată din următorii biți : 0 1 1 0 0 1 1 0 0 1 1 ...

Poziția	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**...**
Bit	0	1	1	0	0	1	1	0	0	1	1	…

Tabelul 4.5. Biții din cheia finală

Emițătorul și *receptorul*, având aceiași *cheie finală*, secretă, unică și sigură, efectuează următorii pași :

Pasul 1. În funcție de biții 1 și 2 din *cheia finală*, stabilesc cele două baze pentru polarizarea fotonilor ce urmează a fi transmiși, respectiv recepționați, conform tabelului nr. 4.6 :

Bit 1	**Bit 2**
0	1

→

Baza 1	**Baza 2**
L (liniar)	C (circular)

Tabelul 4.6. Biții 1 și 2 din *cheia finală – base selection*

Conform tabelului rezultă că vor utiliza pentru polarizarea fotonilor bazele **L** (liniar) și **C** (circular).

Pasul 2. Stabilesc intervalul de timp la care *emițătorul* va transmite un qbit real iar *receptorul* va citi qbitul respectiv.

Bit 3	**Bit 4**	**Bit 5**	**Bit 6**	**Număr$_{10}$**	**Timp (μs)**
1	0	0	1	9	10

Tabelul 4.7. Biții 3,4,5 și 6 din cheia finală – *transmission synchronization*

Emițătorul va transmite fotoni falși tot timpul dar, conform tabelului 4.7, din 10 în 10 microsecunde, va transmite un qbit real pe care *receptorul* îl va citi și nota.

Pasul 3. De la bitul 7 până la terminarea biților din *cheia finală*, fiecare bit care trebuie transmis va fi polarizat conform tabelului nr. 4.8.

Bit din cheia finală	**0**	**1**
Bază polarizare	Baza 1	Baza 2
Polarizare	L	C

Tabelul 4.8. Selectarea bazelor de polarizare a biților

Va rezulta că biții care trebuiesc transmiși vor fi polarizați iar cei care trebuie citiți vor fi interpretați conform tabelului 4.9.

Nr	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**…**
Bit	0	1	1	0	0	1	1	0	0	1	1	…
Baze							C	L	L	C	C	…

Tabelul 4.9. Bazele de polarizare a fotonilor

Astfel, primul bit pe care trebuie să-l transmitem va fi polarizat de către *emițător* respectiv citit de către de către *receptor* în baza de polarizare circulară, al doilea bit va fi transmis de către *emițător* după un interval de 10μs, va fi polarizat în bază liniară, iar receptorul va citi după un interval de 10μs de la recepționarea primului qbit și îl va citi în baza de polarizare liniară și așa mai departe. După ultimul bit din *cheia finală*, procesul de polarizare se va repeta începând cu bitul 7 și așa mai departe.

Pasul 4. Emițătorul și receptorul comunică pe canalul clasic, împart secvența de biți obținută în blocuri egale de biți și compară paritatea fiecărui bloc. În cazul în care paritatea unui bloc de biți diferă, se va relua transmisia blocului respectiv continuând de la pasul 3.

4.6 Pseudocodul protocolului BSTS

Emițătorul și *receptorul*, dețin aceiași *cheie finală*, secretă, unică și sigură, obținută după etapele *Secret key reconciliation* și *Privacy amplification* ale oricărei scheme de distribuire a cheilor cuantice – șirul ***c***.

Etapa 1 - stabilirea bazelor de polarizare – *base selection*

Emițător și ***Receptor:***

```
IF c[1] == 0
    IF c[2] == 0
```

```
        baza1 = L
        baza2 = D
    ELSE            //c[2] = 1
        baza1 = L
        baza2 = C
    ENDIF
ELSEIF              // c[1] = 1
    IF c[2] == 0
        baza1 = C
        baza2 = D
    ELSE            //c[2] = 1
        baza1 = L
        baza2 = D
    ENDIF
ENDIF
```

Etapa 2 - stabilirea tactului transmisiei – *transmission synchronization*

Emițător și ***Receptor:***

t = *convertește* biții [3, 4, 5, 6] din binar în zecimal + 1

Etapa 3 – stabilește polarizarea fiecărui foton in parte

Emițător și ***Receptor:***

```
FOR (i = 7; i <= c.length; i++)
    IF c[i] == 0
        b[j] = baza1
    ELSE    // c[i] = 1
        b[j] = baza2
    ENDIF
ENDFOR
```

Etapa 4 - transmisia poate fi realizată bidirecțional pe canale diferite

Emițător :

```
FOR (k = 0; k <= șir.length; k++)
    alege polarizarea funcție de valoarea bitului șir[k]
    time = 0
    polarizează fotonul în baza b[k] rezultă qbitul p[k]
    transmite qbitul p[k]
    DO
        transmite fotoni falși polarizați aleator
    WHILE (time < t)
ENDFOR
```

Receptor :

recepționează qbitul **p**[k]
wait **t** ms

Etapa 5 – verificarea transmisiei

Emițător și ***Receptor:***

```
FOR (k = 0; k <= șir.length; k++)
   împarte șir în blocuri de 16 biți
   FOR (fiecare bloc de biți calculează paritatea pp)
      transmite pp
      recepționează pp'
      IF pp ≠ pp'
         retransmite blocul
      ENDIF
   ENDFOR
ENDFOR
```

4.7 Diagrama funcțională a protocolului BSTS

Diagrama funcțională a protocolului cuantic de cominicații Base Selection and Transmission Synchronization este prezentat în figura 4.3.

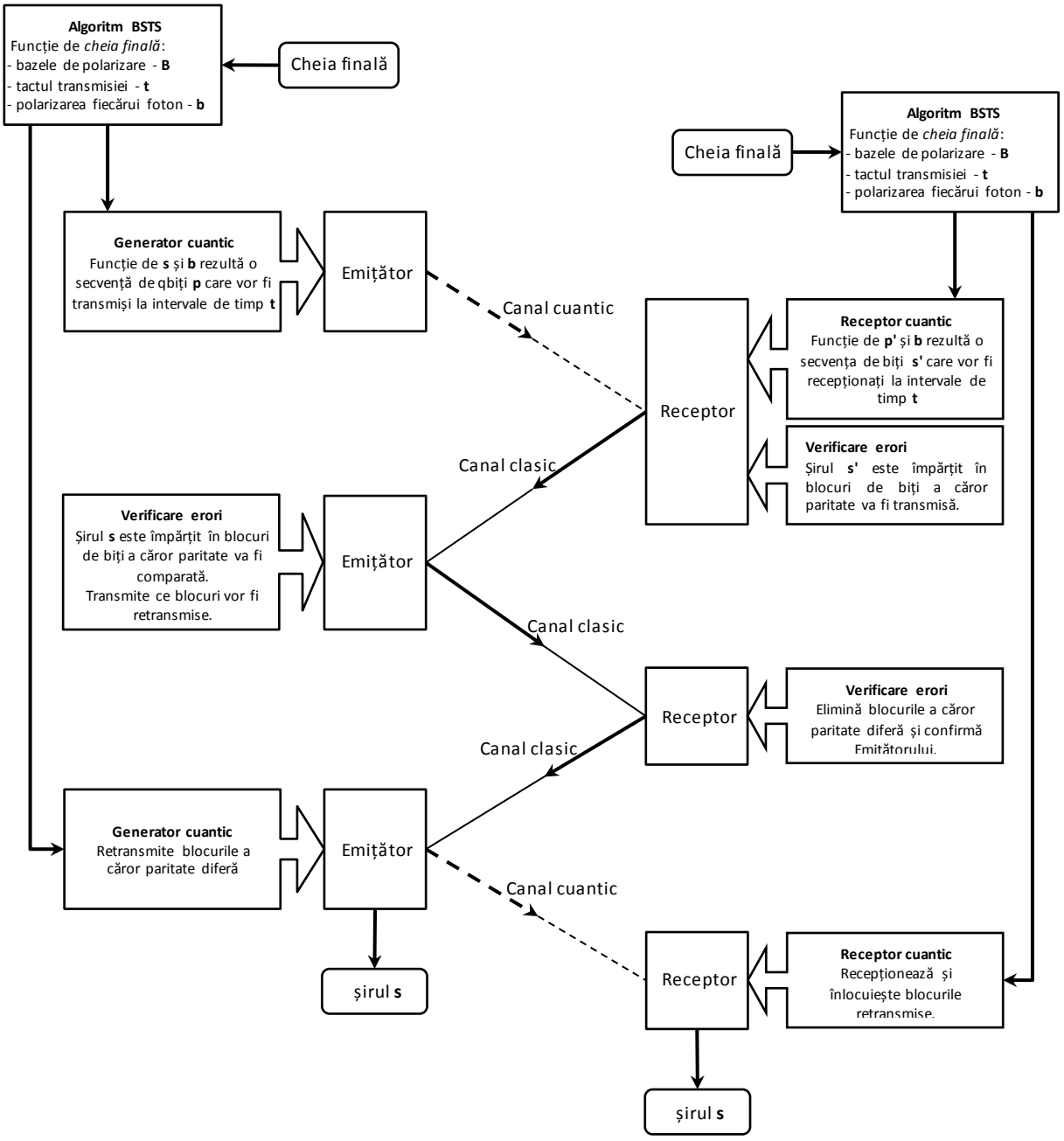

Figura 4.3. Diagrama funcțională a protocolului BSTS

4.8 Avantajele protocolului BSTS

Protocolul Base Selection and Transmission Synchronization utilizează *cheia finală* rezultată după etapele *Secret key reconciliation* și *Privacy amplification*, ale oricărui sistem de distribuire a cheilor cuantice. În funcție de *cheia finală*, *emițătorul* și *receptorul* vor stabili cu exactitate parametrii utilizați în transmisia cuantică, respectiv sistemul de baze de polarizare utilizat, tactul transmisiei și tipul de polarizare aplicat fiecărui foton în parte.

Protocolul BSTS poate fi considerat un protocol de rețea cuantic între două calculatoare deoarece poate realiza o legătură bidirecțională între ele. Prin intermediul protocolului BSTS nu se dorește realizarea doar a unui schimb de chei cuantice, ca în cazul celorlalți algoritmi de distribuire a cheilor cuantice gen BB84, B92 sau E91, ci a se realiza o transmisie bidirecțională între cele două părți care comunică.

Prima etapă a algoritmului BSTS, numită și *Base selection*, stabilește funcție de biții din *cheia finală* cele două baze de polarizare care vor folosite dintre cele trei posibile, *liniar-diagonal*, *liniar-circular* sau *diagonal-circular*, astfel încât avem stabilită prima necunoscută a unei transmisii cuantice. Acest procedeu va micșora considerabil probabilitatea ca *inamicul* să poată descifra transmisia deoarece nu știe ce baze de polarizare să folosească pentru citirea qbiților.

A doua etapă a algoritmului BSTS, numită și *Transmission synchronization*, stabilește funcție de biții din cheia finală intervalul de timp în care *emițătorul* va transmite un foton real iar *receptorul* va citi fotonul respectiv, deoarece în restul intervalului *emițătorul* va transmite fotoni falși polarizați aleatoriu. Acest procedeu va face ca *inamicul* să obțină o cantitatea mult mai mare de qbiți falși decât qbiți reali.

A treia etapă a algoritmului stabilește, pentru fiecare bit care trebuie transmis, ce bază de polarizare i se va aplica astfel încât *emițătorul* și *receptorul* vor ști exact cum să polarizeze fotonul purtător respectiv să citească qbitul respectiv.

A patra etapă a algoritmului BSTS realizează corectarea erorilor apărute în timpul transmisiei cuantice, stabilind ce blocuri de biți trebuie retransmiși.

În concluzie putem spune că algoritmul Base Selection and Transmission Synchronization va realiza o transmisie respectiv recepție, fără erori datorită faptului că *emițătorul* și *receptorul* vor ști exact cum să polarizeze, respectiv să citească fiecare qbit în parte iar la sfârșitul transmisie se va realiza o corectare a erorilor.

4.9 Concluzii și contribuții

4.9.1 Concluzii

În acest capitol am propus un nou protocol criptografic cuantic de comunicații, care folosește biții din *cheia finală*, obținută după etapele *Secret key reconciliation* și *Privacy amplification,* ale oricărui sistem de distribuire a cheilor cuantice și care stabilește cu exactitate parametrii care vor fi utilizați de către *emițător* și *receptor* pentru realizarea transmisiei cuantice.

Datorită faptului că, încă din 1985, au fost descrise principiile de funcționare ale unui calculator cuantic [22], ne putem aștepta ca atunci când va fi operațional, algoritmul one-time pad să poată fi spart prin folosirea metodei de atac *brute force*. Acest lucru va face ca schemele de distribuire a cheilor cuantice să fie inutile, dar folosind algoritmul Base Selection and Transmission Synchronization – BSTS, care realizează o transmisie bidirecțională în totalitate cuantică vom putea comunica în secret.

Dacă în cazul celorlalte sisteme de distribuire a cheilor cuantice cheia finală poate fi de dimensiuni nesatisfăcătoare ca dimensiune, deoarece aceasta trebuie să fie de dimensiunea textului care se dorește a fi criptat, utilizarea aceste chei în cadrul protocolului BSTS este pe deplin satisfăcătoare.

Acest procedeu va face ca *inamicul* să obțină o cantitatea mult mai mare de qbiți falși decât qbiți reali.

4.9.2 Contribuții

Proiectarea, realizarea și implementarea primului protocol de comunicații bidirecțional cuantic între două calculatoare, în care comunicația se realizează exclusiv pe un canal cuantic.

Prin etapa *base selection*, protocolul Base Selection and Transmission Synchronization stabilește cu exactitate ce baze de polarizare vor fi folosite și cum va fi polarizat respectiv citit fiecare qbit în parte.

Prin etapa *transmission synchronization*, protocolul Base Selection and Transmission Synchronization stabilește intervalul de timp în care *emițătorul* va transmite un foton real iar *receptorul* va citi fotonul respectiv, deoarece în restul

intervalului *emițătorul* va transmite fotoni falși polarizați aleatoriu astfel încât *inamicul* să obțină mai mulți qbiți falși decât qbiți reali.

Aceste contribuții sunt baza următoarelor lucrări științifice :

- Anghel C., „Quantum cryptography algorithm", Proceedings ECIT2008 – 5th European Conference on Intelligent Systems and Technologies, Romania, Iasi, 2008.
- Anghel C. & Coman G., „Base selection and transmission synchronization algorithm in quantum cryptography", Proceedings CSCS17 - 17th International Conference on Control Systems and Computer Science, Romania, Bucharest, ISSN : 2066-4451, vol. 1, pag. 281-284, 2009.

5. Proiectarea metodei Quantum Bit Travel Time de detectare a inamicului

5.1 Introducere

Metodele actuale de detectare a inamicului care interceptează un canal cuantic sunt ineficiente pentru unele metode de atac cum ar fi Photon Number Splitting, Man-In-the-Middle sau Quantum Cloning.

În acest capitol am propus o nouă metodă de detectare a interceptării care poate detecta prezența inamicului indiferent de metoda de atac folosită. Această metodă se va numi Quantum Bit Travel Time – QBTT [5] și va realiza o detectare imediată și mai ales exactă a inamicului.

Metoda Quantum Bit Travel Time poate fi implementată în oricare sistem de distribuire a cheilor cuantice și are avantajul că *inamicul* poate fi detectat de către *receptor* în timpul transmisiei cuantice, după fiecare qbit recepționat.

5.2 Necesitatea metodei

Securității sistemelor de distribuire a cheilor cuantice poate fi îmbunătățită prin detectarea „exactă" și „imediată" a *inamicului*.

Detectarea „exactă" a *inamicului* se referă la faptul că acesta nu trebuie să fie confundat cu perturbările cauzate de canalul cuantic de comunicații.

Detectarea „imediată" a *inamicului* se referă la faptul că acesta trebuie detectat în momentul în care a intervenit pentru a citi un qbit. În cazul celorlalți algoritmilor de distribuire a cheilor cuantice, detectarea *inamicului* se face după finalizarea transmisiei cuantice, după etapa de comunicare a bazelor de polarizare pe canalul clasic, existând astfel posibilitatea ca *inamicul* să obțină prea multe informații legate de cheia transmisă.

Metoda Quantum Bit Travel Time – QBTT de detectare a *inamicului* poate fi implementată în oricare sistem de distribuire a cheilor cuantice și are avantajul că *inamicul* poate fi detectat cu *exactitate* și mai ales *imediat* după ce acesta a intervenit pentru a citi un qbit, în timpul transmisiei cuantice.

5.3 Premisele metodei QBTT

Metoda propusă se bazează pe faptul că filtrele de polarizare induc întârzieri ale particulei purtătoare [77]. Fiecare filtru aplicat va întârzia particula purtătoare cu o anumită perioadă de timp.

Filtrele de polarizare folosite de către *emițător* și *receptor* pentru transformarea biților în qbiți, respectiv a qbiților în biți, induc întârzieri ale particulei purtătoare. Vom nota cu ΔT această întârziere care va fi identică pentru fiecare particulă purtătoare. Această întârziere ΔT se va determina experimental, pentru fiecare tip de schemă folosită, ea fiind condiționată de tipul filtrelor de polarizare utilizate de către *emițător* și *receptor*.

Emițătorul și *receptorul* vor sincroniza ceasurile interne ale dispozitivelor de lucru [20,32,38]. *Emițătorul*, pentru fiecare qbit emis, generează și transmite receptorului, pe canalul clasic, timestamp-ul momentului emisiei, la rândul său *receptorul* pentru fiecare qbit recepționat generează timestamp-ul momentului recepției, diferența dintre cele dintre cele două timestamp-uri, timestamp_receptor - timestamp_emițător, va reprezenta întârzierea ΔT a particulei purtătoare datorate celor două filtre de polarizare aplicate de *emițător* și *receptor*.

Dacă *inamicul* citește un qbit, adică aplică un filtru de polarizare, acesta va induce o întârziere suplimentară a particulei purtătoare, notată Δt, crescând astfel timpul parcurs de către aceasta de la *receptor* la *emițător*.

Inamicul poate fi detectat prin măsurarea timpului parcurs de particula purtătoare de la *emițător* la *receptor*, iar dacă $\Delta T' = \Delta T + \Delta t$ va rezulta că *inamicul* a fost prezent.

Alt avantaj al metodei QBTT rezultă din faptul că se pot face verificări pentru detectarea *inamicului* în timpul transmisiei cuantice, după fiecare qbit transmis de la *emițător* către *receptor* față de celelalte metode la care detectarea *inamicului* se face la sfârșitul transmisiei cuantice, după comunicarea bazelor de polarizare pe canalul clasic.

Noua metodă poate fi implementată cu ușurință în orice sistem de distribuire a cheilor cuantice ca o măsură suplimentară de securitate sau ca metodă de sine stătătoare de detectare a *inamicului*.

5.4 Implementarea metodei QBTT în sistemul BB84

Implementarea noii metode de detectare a inamicului în sistemul de distribuire a cheilor cuantice BB84 cuprinde următoarele etape:

1. *Emițătorul* și *receptorul* își sincronizează ceasurile interne.
2. *Emițătorul* generează o secvență aleatoare de biți - notată "**s**".
3. *Emițătorul* alege aleator ce bază de polarizare să aplice fiecărui foton din "**s**" (liniar "L" sau diagonal "D"). Notăm "**b**" secvența de polarizare.
4. *Emițătorul*, folosind un set de filtre de polarizare, creează o secvență "**p**" de qbiți, a căror direcție de polarizare reprezintă biții din "**s**".
5. *Emițătorul* transmite qbiții din "**p**" prin fibră optică către *receptor*. Pentru fiecare qbit transmis, *emițătorul* transmite *receptorului* pe canalul clasic momentul de timp în care a fost transmis – timestamp_emițător.
6. *Receptorul*, pentru fiecare qbit recepționat, alege aleator câte o bază de polarizare (liniar "L" sau diagonal "D"). Notăm cu "**b'**" secvența bazelor de polarizare aleasă.
7. Pentru fiecare qbit recepționat, *receptorul* generează timestamp_receptor și calculează întârzierea ΔT = timestamp_receptor - timestamp_emițător, dintre momentul transmisiei și momentul recepției. Dacă pentru un anumit qbit întârzierea ΔT este mai mare decât cea normală atunci înseamnă că *inamicul* este prezent și oprește transmisia.
8. *Receptorul*, măsoară fiecare qbit recepționat respectând baza de polarizare aleasă la pct. 6, rezultând o secvență de biți "**s'**".
9. *Emițătorul* îi transmite printr-un canal public *receptorului*, ce bază de polarizare a ales pentru fiecare bit în parte. La rândul sau *receptorul* îi comunică *emițătorului* unde a făcut aceeași alegere a bazei de polarizare. Biții pentru care cei doi nu au avut aceeași bază de polarizare sunt eliminați din "**s**" și "**s'**".

5.5 Detectarea inamicului în sistemul BB84 cu metoda QBTT

Emițătorul, pentru fiecare qbit emis, generează și transmite receptorului, pe canalul clasic, timestamp-ul momentului emisiei, la rândul său *receptorul* pentru fiecare qbit recepționat generează timestamp-ul momentului recepției, diferența

dintre cele dintre cele două timestamp-uri, timestamp_receptor - timestamp_emițător va reprezenta întârzierea ΔT a particulei purtătoare datorate celor două filtre de polarizare aplicate la emisie respectiv recepție.

Dacă inamicul încercă să citească un anumit qbit, el va induce o nouă întârziere Δt, rezultând o întârziere totală a qbitului $\Delta T' = \Delta T + \Delta t$. Aceasta îi va permite receptorului să identifice prezența inamicului și să oprească transmisia.

5.6 Pseudocodul unui sistem BB84 cu metoda QBTT

Pseudocodul sistemului de distribuire a a cheilor cuantice BB84 cu metoda QBTT de detectare a inamicului este :

Pasul 1 – comunicarea pe canalul cuantic

Emițătorul:

generează aleatoriu dintre (0,1) șirul ***s***
FOR fiecare bit din ***s***
    alege aleatoriu dintre ("L", "D") rezultând baza **b**[i]
ENDFOR
FOR fiecare bit din ***s***
    generează un foton
    IF ***s***[i] = 1 și **b**[i] = L
        THEN *polarizează* fotonul în starea (90^o) rezultă un qbit **p**[i] = ↑
    IF ***s***[i] = 0 și **b**[i] = L
        THEN *polarizează* fotonul în starea (0^o) rezultă un qbit **p**[i] = →
    IF ***s***[i] = 1 și **b**[i] = D
        THEN *polarizează* fotonul în starea (135^o) rezultă un qbit **p**[i] = ↖
    IF ***s***[i] = 0 și **b**[i] = D
        THEN *polarizează* fotonul în starea (45^o) rezultă un qbit **p**[i] = ↗
    generează timestamp ***T***
    trimite ***T*** către ***Receptor*** // canalul clasic
    trimite qbitul **p**[i] către ***Receptor*** // canalul cuantic
ENDFOR

Receptor :

FOR fiecare qbit **p'**[i] recepționat
    generează aleatoriu dintre ("L", "D") rezultând baza **b'**[i]
    măsoară qbitul **p'**[i] în baza **b'**[i] rezultă bitul ***s'***[i]
    generează timestamp ***T'***
    calculează $\Delta T = T' - T$
    IF ΔT NU este în limite normale

Inamicul *a fost prezent*
oprește *transmisia*
ENDIF
ENDFOR

Pasul 2 – comunicarea pe canalul clasic

Receptor:
FOR fiecare bit ***s'***[i]
transmite baza **b'**[i] către ***Emițător***
ENDFOR

Emițător:
FOR fiecare bit ***s'***[i]
transmite baza **b**[i] către ***Receptor***
ENDFOR

Pasul 3 – Bases reconciliation

Receptor:
FOR fiecare bit ***s'***[i]
IF baza **b'**[i] ≠ **b**[i]
elimină ***s'***[i] din șirul ***s'***
ENDIF
ENDFOR

Emițător:
FOR fiecare bit ***s***[i]
IF baza **b**[i] ≠ **b'**[i]
elimină ***s***[i] din șirul ***s***
ENDIF
ENDFOR

5.7 Diagrama funcțională a sistemului BB84 cu metoda QBTT

Diagrama funcțională a unui sistem BB84 cu metoda QBTT de detectare a inamicului este prezentată în figura 5.1.

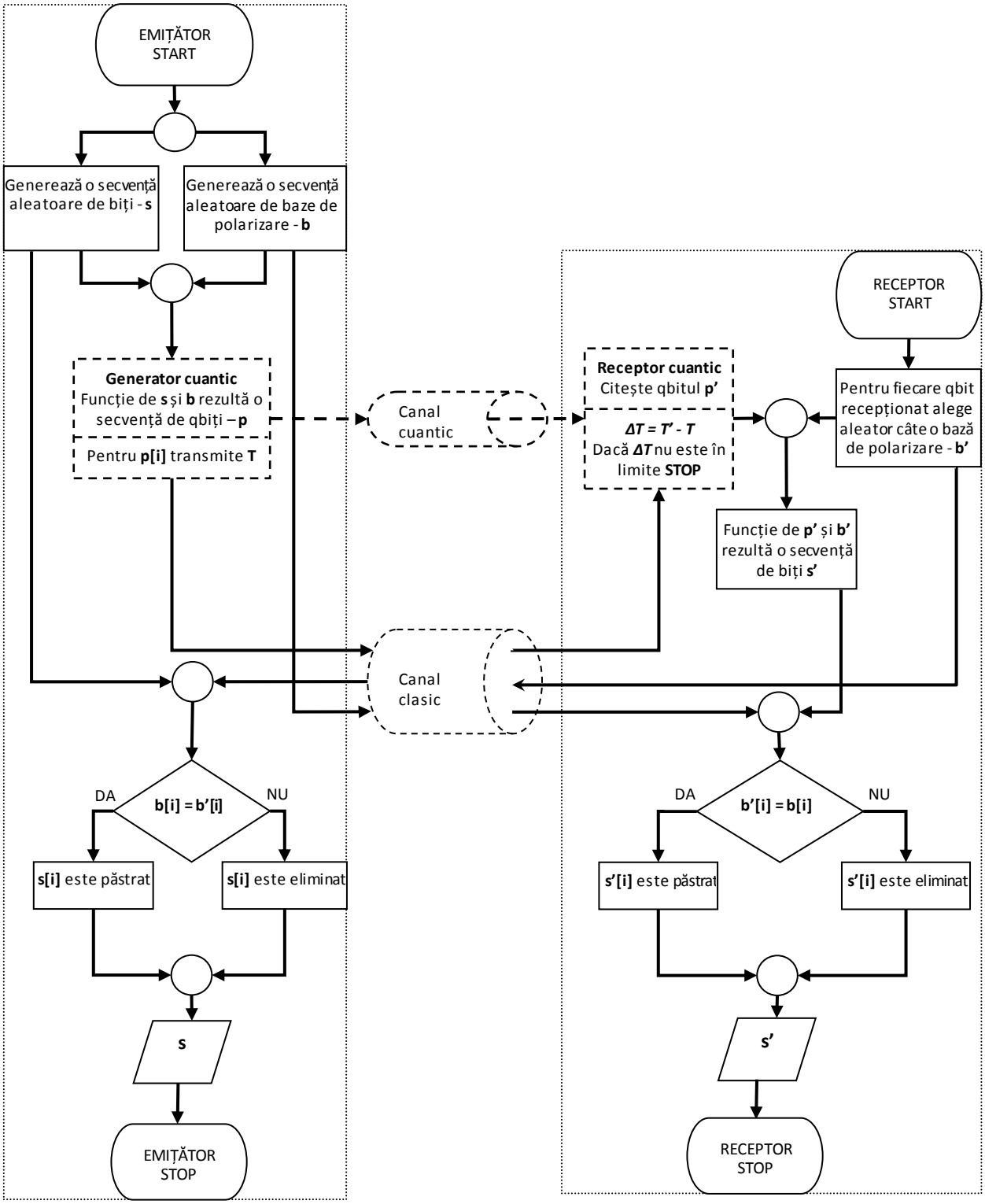

Figura 5.1. Diagrama funcțională a unui sistem BB84 cu metoda QBTT

5.8 Implementarea metodei QBTT în sistemul B92

Implementarea noii metode de detectare a inamicului în sistemul de distribuire a cheilor cuantice B92 cuprinde următoarele etape:

1. *Emițătorul* și *receptorul* își sincronizează ceasurile interne.
2. *Emițătorul* generează o secvență aleatoare de biți - notată "***s***".
3. *Emițătorul* utilizează stările de polarizare (0^o, 45^o) pentru a reprezenta biții din "***s***". Notăm "**b**" secvența de polarizare.
4. *Emițătorul*, folosind un echipament special, creează o secvență "**p**" de fotoni polarizați, conform tabelului 2, a căror direcție de polarizare reprezintă biții din "***s***".
5. *Emițătorul* transmite qbiții din "***p***" prin fibră optică către *receptor*. Pentru fiecare qbit transmis, ***emițătorul*** transmite *receptorului* pe canalul clasic momentul de timp în care a fost transmis – timestamp_emițător.
6. *Receptorul*, pentru fiecare qbit recepționat, alege aleator câte o bază de polarizare (liniar "L" sau diagonal "D"). Notăm cu "**b'**" secvența bazelor de polarizare aleasă.
7. Pentru fiecare qbit recepționat, *receptorul* generează timestamp_receptor și calculează întârzierea ΔT = timestamp_receptor - timestamp_emițător, dintre momentul transmisiei și momentul recepției. Dacă pentru un anumit qbit întârzierea ΔT este mai mare decât cea normală atunci înseamnă că *inamicul* este prezent și oprește transmisia.
8. *Receptorul*, măsoară fiecare qbit recepționat în parte, respectând bazele de polarizare alese la pct. 5 și îi comunică printr-un canal public ***emițătorului*** unde a primit un '0'.

   Dacă rezultatul va fi un '0', atunci ***s'*** = ? iar bitul respectiv va fi eliminat din ***s'***.

   Dacă rezultatul va fi un '1', atunci ***s'*** = 0 dacă **b'** = D și

   s' = 1 dacă **b'** = L
9. *Emițătorul* elimină din ***s*** biții unde a primit un '0' de la *receptor*.

5.9 Detectarea inamicului în sistemul B92 cu metoda QBTT

Emițătorul, pentru fiecare qbit emis, generează și transmite receptorului, pe canalul clasic, timestamp-ul momentului emisiei, la rândul său *receptorul* pentru fiecare qbit recepționat generează timestamp-ul momentului recepției, diferența dintre cele dintre cele două timestamp-uri, timestamp_receptor - timestamp_emițător va reprezenta întârzierea ΔT a particulei purtătoare datorate celor două filtre de polarizare aplicate la emisie respectiv recepție.

Dacă inamicul încercă să citească un anumit qbit, el va induce o nouă întârziere Δt, rezultând o întârziere totală a qbitului $\Delta T' = \Delta T + \Delta t$. Aceasta îi va permite receptorului să identifice prezența inamicului și să oprească transmisia.

5.10 Pseudocodul unui sistem B92 cu metoda QBTT

Pseudocodul sistemului de distribuire a a cheilor cuantice B92 cu metoda QBTT de detectare a inamicului este :

Pasul 1 – comunicarea pe canalul cuantic

Emițător:

generează aleatoriu dintre (0,1) șirul ***s***
FOR fiecare bit din **s**
    generează un foton
    IF **s**[i] = 0 THEN *polarizează* fotonul în starea (0^o) rezultă un qbit **p**[i] = →
    IF **s**[i] = 1 THEN *polarizează* fotonul în starea (45^o) rezultă un qbit **p**[i] = ↗
    generează timestamp ***T***
    trimite ***T*** către ***Receptor*** // canalul clasic
    trimite qbitul **p**[i] către ***Receptor*** // canalul cuantic
ENDFOR

Receptor:

FOR fiecare qbit **p'**[i] recepționat
    generează aleatoriu dintre ("L", "D") rezultând baza **b'**[i]
    măsoară qbitul **p'**[i] în baza **b'**[i] rezultând o valoare **v** // unde $v \in \{0,1\}$
    generează timestamp ***T'***
    calculează $\Delta \boldsymbol{T} = \boldsymbol{T'} - \boldsymbol{T}$
    IF $\Delta \boldsymbol{T}$ NU este în limite
        Inamicul *a fost prezent*
        oprește *transmisia*
    ENDIF

```
    IF v = 0
       s'[i] = ?
       elimină bitul corespondent din s'
       trimite valoarea '0' către Emițător // pe canalul clasic
    ELSE                    // v = 1
       IF b'[i] = D
          THEN s'[i] = 0
       ELSE                 // b'[i] = R
          THEN s'[i] = 1
       ENDIF
    ENDIF
 ENDFOR
```

Pasul 2 – comunicarea pe canalul clasic

Emițător:

```
FOR fiecare valoare '0' recepționată de Receptor
   elimină bitul corespondent din s
ENDFOR
```

5.11 Diagrama funcțională a sistemului B92 cu metoda QBTT

Diagrama funcțională a unui sistem B92 cu metoda QBTT de detectare a inamicului este prezentată în figura 5.2.

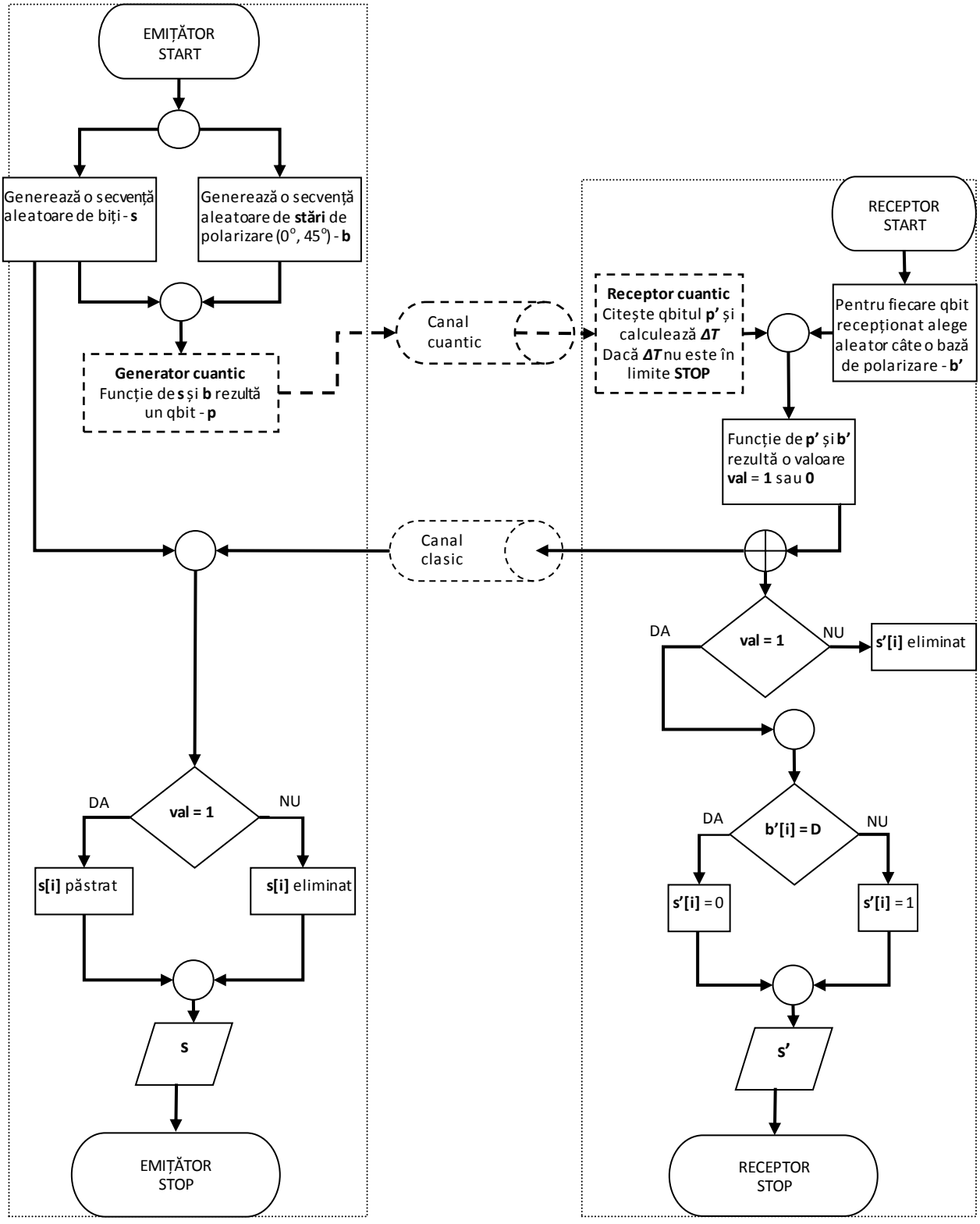

Figura 5.2. Diagrama funcțională a unui sistem B92 cu metoda QBTT

5.12 Avantajele metodei QBTT

Principalele avantaje al metodei Quantum Bit Travel Time – QBTT sunt acelea că *inamicul* poate fi detectat indiferent de metoda de atac folosită, este detectat „imediat" și cu „exactitate".

Inamicul este detectat indiferent de metoda de atac folosită deoarece orice metodă de atac presupune o întârziere a particulei purtătoare iar această întârziere va alerta *receptorul* de prezența *inamicului*.

Receptorul poate detecta „imediat" prezența *inamicului* în timpul transmisiei cuantice, după fiecare qbit recepționat, prin măsurarea timpului parcurs de particula purtătoare de la *emițător* la *receptor*. În cazul altor algoritmilor de distribuire a cheilor cuantice, detectarea *inamicului* se face după finalizarea transmisiei cuantice, în etapa de comunicare a bazelor de polarizare pe canalul clasic, lăsând astfel în mâna *inamicului* informații importante legate de transmisie.

Inamicul poate fi depistat cu „exactitate" deoarece perturbările naturale apărute pe canalul cuantic nu produc întârzieri ale particulei purtătoare ci doar modificări ale polarizării. Această proprietate ne ajută să nu confundăm eventualele zgomotele de pe canalul cuantic cu prezența unui *inamic*. Celelalte metode de depistare a inamicului nu pot face diferența între zgomot de e canalul cuantic și prezența unui eventual *inamic*.

Un alt avantaj este acela că această metodă poate fi implementată cu ușurință în orice tip de schemă de distribuire a cheilor cuantice.

5.13 Concluzii și contribuții

5.13.1 Concluzii

În acest capitol am propus o nouă metodă, *exactă* și *rapidă*, de detectare a unui eventual *inamic* care interceptează canalul cuantic de comunicații dintre *emițător* și *receptor*.

Am văzut, în capitolul 3, că există anumite metode de atac, metode pur teoretice deocamdată, în care *inamicul*, chiar dacă interceptează transmisia, nu poate fi detectat prin metodele actuale și care pot compromite în totalitate transmisia cuantică dintre *emițător* și *receptor*. În schimb, prin utilizarea metodei Quantum Bit

Travel Time, *inamicul* poate fi detectat imediat ce a intervenit pentru a citi un qbit, indiferent de metoda de atac folosită.

Datorită faptului că nou metodă propusă poate detecta prezența inamicului în timpul transmisiei cuantice, verificare făcându-se după fiecare qbit recepționat, iar inamicul este detectat cu exactitate, fără dubii că ar putea fi confundat cu erorile ale transmisiei de pe canalul cuantic, elimină în totalitate riscul ca *inamicul* să poată intercepta transmisia cuantică fără a fi detectat, indiferent de metoda de atac folosită de către acesta.

În concluzie putem spune că metoda Quantum Bit Travel Time - QBTT de detectare a inamicului poate fi implementată în oricare sistem de distribuire a cheilor cuantice și are avantajul că *inamicul* poate fi detectat cu *exactitate* și mai ales *imediat* după ce acesta a intervenit pentru a citi un qbit, în timpul transmisiei cuantice.

5.13.2 Contribuții

Proiectarea, realizarea și implementarea unei metode de detectare a interceptării care poate detecta prezența inamicului indiferent de metoda de atac folosită. Această metodă va realiza o detectare imediată și mai ales exactă a inamicului.

Detectarea „exactă" a *inamicului* se referă la faptul că acesta nu este confundat cu perturbările cauzate de canalul cuantic de comunicații.

Detectarea „imediată" a *inamicului* se referă la faptul că acesta este detectat în momentul în care a intervenit pentru a citi un qbit.

Această metodă de detectare a interceptării poate fi implementată în orice sistem de distribuție a cheilor cuantice.

Aceste contribuții sunt baza următoarelor lucrări științifice :

- Anghel C., „Creșterea securității Sistemelor Informatice și de Comunicații prin Criptografia Cuantică", Cornell University Library, http://arxiv.org/, cite as: http://arxiv.org/abs/1006.5381v1, 2010.
- Anghel C., „New eavesdropper detection method in quantum cryptography", The annals of "Dunarea de Jos" University of Galati, fascicula III, vol. 34, nr. 1, pag. 1-8, 2011.

6. Proiectarea protocolului cuantic Base Selection and Polarization Agreement

6.1 Introducere

Acest capitol propune un nou protocol cuantic de comunicații, având la bază protocolul Base Selection and Transmission Synchronization [3] îmbunătățit cu ajutorul metodei Quantum Bit Travel Time de detectare a inamicului [5] care se va numi Base Selection and Polarization Agreement – BSPA [6].

Prin implementarea metodei QBTT de detectare a inamicului în protocolul BSTS, inamicul poate fi detectat în timpul transmisiei cuantice și fără dubii că ar putea fi confundat cu perturbări ale transmisiei. Acest lucru ne permite să simplificăm metoda BSTS prin eliminarea etapei *Transmission Synchronization* adică prin eliminarea intervalului de timp prestabilit la care se face transmisia cuantică, rezultând astfel un protocol de comunicații cuantic mult mai rapid.

6.2 Premisele protocolului BSPA

Parametrii vizați în cazul protocolului cuantic BSPA sunt:

(1) ce pereche de baze de polarizare dintre cele trei, L, D și C, va fi folosită;

(2) ce bază de polarizare va fi folosită pentru codificarea fiecărui bit în parte.

(3) detectarea *inamicului* prin monitorizarea timpului parcurs de către fiecare particulă purtătoare de la *emițător* la *receptor*.

(4) verificarea biților recepționați la finalul transmisiei cuantice și corectarea erorilor prin retransmiterea blocurilor care conțin inadvertențe.

Pentru codificarea biților vom stabili următoarea convenție de codificare și reprezentare a qbiților :

Baza	**L**	**D**	**C**	**C**	**L**	**D**
Polarizare	0^o	45^o	spinL	spinR	90^o	135^o
Qbit	↔	↗	↺	↻	↕	↘
Bit	0	0	0	1	1	1

Tabelul 6.1. Polarizarea fotonilor - qbiți

Prin protocolul cuantic de comunicații BSPA, *emițătorul* și *receptorul* vor stabili în comun, în funcție de biții din *cheia finală*, care pereche, dintre cele trei baze de polarizare, *liniar-diagonal*, *liniar-circular* sau *diagonal-circular* vor fi folosite pentru polarizarea fotonilor.

În funcție de biții rămași din *cheia finală*, *emițătorul* și *receptorul*, vor ști exact ce baze de polarizare să aplice pentru fiecare foton pe care urmează să-l transmită respectiv să-l recepționeze.

În timpul transmisie cuantice, după fiecare qbit recepționat, *receptorul* va verifica dacă intervalul de timp parcurs de particula purtătoare, de la *emițător* la *receptor* se încadrează în limitele normale. În cazul în care intervalul de timp este mai mare decât normalul înseamnă că un *inamic* a interceptat particula purtătoare și oprește transmisia.

După terminarea transmisiei cuantice, *emițătorul* și *receptorul*, împart secvența de biți obținută în blocuri de biți și vor compara paritatea fiecărui bloc. În cazul în care paritatea unui bloc de biți diferă, se va relua transmisia blocului respectiv.

Schema bloc a unui sistem cuantic de comunicații bazat pe protocolul BSPA este prezentată în figura 6.1.

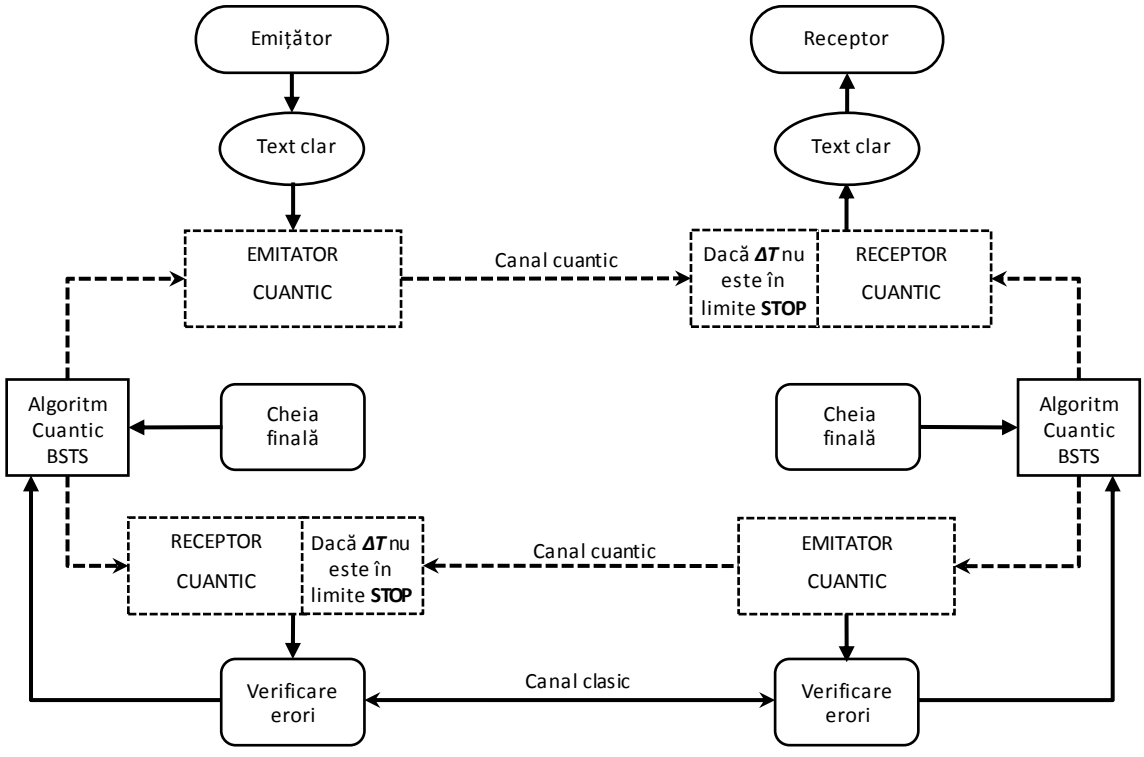

Figura 6.1. Schema bloc a protocolului cuantic de comunicații BSPA

6.3 Implementarea protocolul BSPA

Protocolul va fi executat simultan de către *emițător* și *receptor* și va utiliza biții din *cheia finală* obținută după etapele *Secret key reconciliation* și *Privacy amplification* ale oricărui sistem de distribuire a cheilor cuantice :

1. Primii doi biți din *cheia finală* vor fi utilizați pentru a stabili bazele de polarizare, care vor fi folosite pentru a polariza fotonii ce urmează a fi transmiși, conform tabelului :

Bitul 1	**Bitul 2**	**Baza 1**	**Baza 2**
0	0	L	D
0	1	L	C
1	0	C	D
1	1	L	D

Tabelul 6.2. Selectarea bazelor în funcție de biții 1 și 2 din *cheia finală – base selection*

2. Biții rămași din *cheia finală* vor reprezenta bazele de polarizare pentru fiecare bit care va fi transmis, respectiv recepționat, conform tabelului 6.3 :

Dacă Bitul este	**0**	**1**
Baza de polarizare	Baza 1	Baza 2

Tabelul 6.3. Stabilirea bazelor pentru polarizarea fiecărui bit

3. După ultimul bit din *cheia finală*, procesul de polarizare al fotonilor va continua de la bitul numărul 3 din *cheia finală* până la terminarea mesajului care se dorește a fi transmis.
4. Pentru fiecare qbit transmis, *emițătorul* transmite *receptorului* pe canalul clasic momentul de timp în care a fost transmis – timestamp_emițător. Pentru fiecare qbit recepționat, *receptorul* generează timestamp_receptor și calculează întârzierea ΔT = timestamp_receptor - timestamp_emițător, dintre momentul transmisiei și momentul recepției. Dacă pentru un anumit qbit întârzierea ΔT este mai mare decât cea determinată experimenta atunci înseamnă că *inamicul* este prezent și oprește transmisia.

5. *Emițătorul și receptorul* comunică pe canalul clasic și împart secvența de biți obținută în blocuri de biți comparând paritatea fiecărui bloc. În cazul în care paritatea unui bloc de biți diferă, se va relua transmisia blocului respectiv.

6.4 Pseudocodul protocolului BSPA

Emițătorul și *receptorul*, dețin aceiași *cheie finală*, secretă, unică și sigură, obținută după etapele *Secret key reconciliation* și *Privacy amplification* ale oricărei scheme de distribuire a cheilor cuantice– șirul ***c***.

Etapa 1 - stabilirea bazelor de polarizare – *base selection*

Emițător și ***Receptor:***

```
IF c[1] == 0
    IF c[2] == 0
       baza1 = L
       baza2 = D
    ELSE            //c[2] = 1
       baza1 = L
       baza2 = C
    ENDIF
ELSEIF              // c[1] = 1
    IF c[2] == 0
       baza1 = C
       baza2 = D
    ELSE            //c[2] = 1
       baza1 = L
       baza2 = D
    ENDIF
ENDIF
```

Etapa 2 – stabilește polarizarea fiecărui foton in parte

Emițător și ***Receptor:***

```
FOR (i = 3; i <= c.length;  i++)
   IF c[i] == 0
      b[j] = baza1
   ELSE   // c[i] = 1
      b[j] = baza2
   ENDIF
ENDFOR
```

Etapa 3 - transmisia poate fi realizată bidirecțional pe canale diferite

Emițător :

FOR (k = 0; k <= **șir**.length; k++)
alege polarizarea funcție de valoarea bitului **șir**[k]
polarizează fotonul în baza **b**[k] rezultă qbitul **p**[k]
generează timestamp ***T***
trimite ***T*** către ***Receptor*** // canalul clasic
transmite qbitul **p**[k] către ***Receptor*** // canalul cuantic
ENDFOR

Receptor :

DO
recepționează qbitul **p**[k]
generează timestamp ***T'***
calculează $\Delta T = T' - T$
IF ΔT NU este în limite
Inamicul *a fost prezent*
oprește *transmisia*
ENDIF
WHILE (transmission end)

Etapa 4 – verificarea transmisiei

Emițător și ***Receptor:***

FOR (k = 0; k <= ***șir***.length; k++)
împarte ***șir*** în blocuri de 16 biți
FOR (fiecare bloc de biți calculează paritatea ***pp***)
transmite ***pp***
recepționează ***pp'***
IF ***pp*** ≠ ***pp'***
retransmite blocul
ENDIF
ENDFOR

6.5 Diagrama funcțională a protocolului BSPA

Diagrama funcțională a protocolului Base Selection and Polarization Agreement este prezentată în figura 6.2.

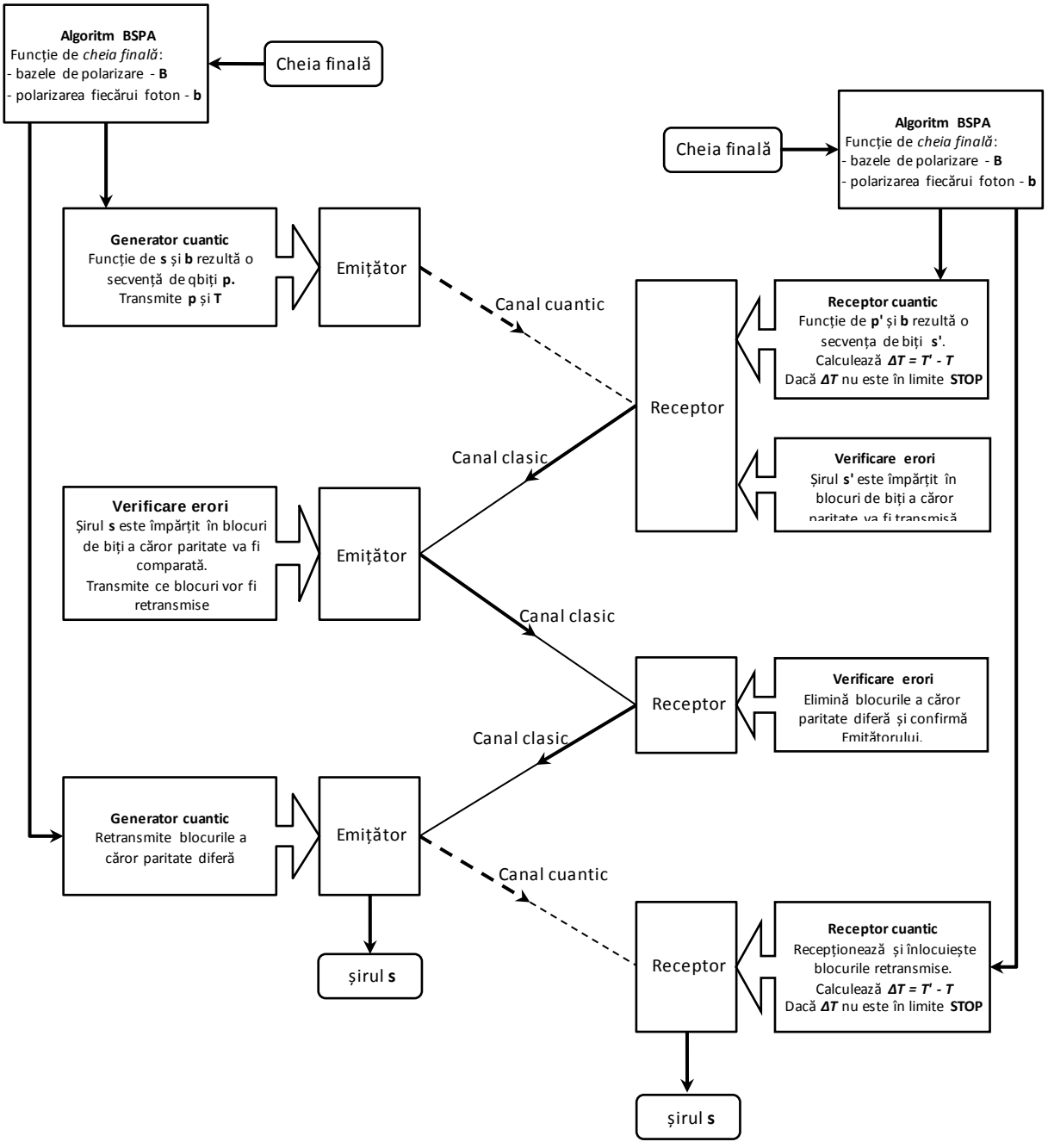

Figura 6.2. Diagrama funcțională a protocolului BSPA

6.6 Avantajele protocolului BSPA

Prin implementarea metodei Quantum Bit Travel Time de detectare a inamicului în protocolul cuantic de comunicații Base Selection and Transmission Synchronization obținem protocolul cuantic de comunicații Base Selection and Polarization Agreement în care inamicul poate fi detectat în timpul transmisiei cuantice, după fiecare qbit recepționat și fără dubii că ar putea fi confundat cu perturbări de pe canalului cuantic deoarece zgomotele de pe canalul cuantic nu produc întârzieri ale particulei purtătoare.

Protocolul BSPA este mai simplu, mai sigur și mai rapid, datorită implementării metodei Quantum Bit Travel Time de detectare a inamicului și a eliminării etapei *Transmission Synchronization*.

Având în vedere faptul că un eventual *inamic* este practic depistat imediat ce încearcă să intercepteze transmisia cuantică, algoritmul BSPA realizează un protocol cuantic de comunicații care poate fi implementat pentru a realiza o rețea cuantică de comunicații.

6.7 Concluzii și contribuții

6.7.1 Concluzii

Utilizarea metodei Quantum Bit Travel Time de detectare a inamicului în coroborare cu protocolul cuantic de comunicații Base Selection and Transmission Synchronization duce la obținerea unui nou protocol cuantic de comunicații numit Base Selection and Polarization Agreement - BSPA, în care inamicul poate fi depistat în timpul transmisiei cuantice, după fiecare qbit recepționat și fără dubii că ar putea fi confundat cu perturbări ale transmisiei deoarece zgomotele de pe canalul cuantic nu produc întârzieri ale particulei purtătoare.

Spre diferență de ceilalți algoritmi cuantici care au ca obiectiv doar schimbul cuantic de chei de criptare utilizate în combinație cu algoritmul de one-time pad, algoritmul BSPA realizează o transmisie bidirecțională în totalitate cuantică, rezultând un algoritm cuantic de comunicații între un emițător și un receptor.

Având în vedere faptul că un eventual *inamic* este practic depistat imediat ce încearcă să intercepteze transmisia cuantică, algoritmul BSPA realizează un protocol

cuantic de comunicații care poate fi implementat pentru a realiza o rețea cuantică de comunicații.

În concluzie putem spune că implementând metoda de detectare a inamicului QBTT în protocolul BSTS obținem un nou protocol, protocolul BSPA, bidirecțional cuantic de comunicații care realizează corectarea erorilor și stabilește parametrii transmisiei.

6.7.2 Contribuții

Proiectarea, realizarea și implementarea unui protocol de comunicații bidirecțional cuantic între două calculatoare, în care comunicația se realizează exclusiv pe un canal cuantic iar prezența inamicului este detectată imediat și cu exactitate indiferent de metoda de atac folosită.

Detectarea imediată a *inamicului* se referă la faptul că acesta este detectat în momentul în care a intervenit pentru a citi un qbit.

Detectarea exactă a *inamicului* se referă la faptul că acesta nu este confundat cu perturbările cauzate de canalul cuantic de comunicații.

Aceste contribuții sunt baza următoarelor lucrări științifice :

- Anghel C., „New eavesdropper detection method in quantum cryptography", The annals of "Dunarea de Jos" University of Galati, fascicula III, vol.34, nr. 1, pag. 1-8, 2011.
- Anghel C., „New quantum cryptographic protocol", The annals of "Dunarea de Jos" University of Galati, fascicula III, vol.34, nr. 2, pag. 7-13, 2011.

7. Simularea sistemelor de distribuire a cheilor cuantice

7.1 Introducere

Datorită lipsei materiale a unor dispozitive practice de polarizare și transmitere a qbiților, respectiv de citire și recepție a fotonilor polarizați, am realizat, ca și alții care au studiat criptografia cuantică [56], o serie de programe care simulează sistemele de distribuire a cheilor cuantice și comportamentul *emițătorului*, *receptorului* și a *inamicului* [7].

Programele sunt scrise în limbajul C++ iar schema bloc a configurației hardware este prezentată în figura 7.1.

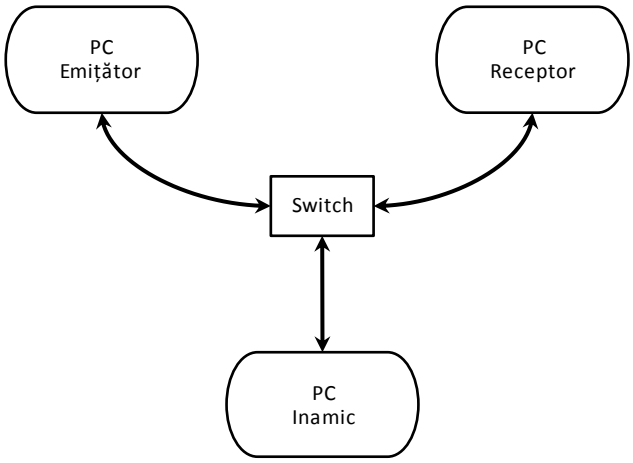

Figura 7.1. Schema bloc a programului de simulare

Pentru reprezentarea qbiților, în programele de simulare, am folosit convenția stabilită în tabelul 7.1.

Baza	Bit	Qbit	Simbol
L	0	→	a
	1	↑	b
D	0	↖	c
	1	↗	d
C	0	↻	e
	1	↺	f

Tabelul 7.1. Reprezentarea biților în programele de simulare

7.2 Simularea sistemului BB84 în condiții ideale fără inamic

7.2.1 Implementarea software

Pentru implementarea programului de simulare BB84-Ideal am folosit limbajul C++. *Emițătorul* și *Receptorul* vor comunica prin canalul cuantic și cel clasic fără prezența *Inamicului* iar legătura între ei se va face prin intermediul unui switch.

Această aplicație software este formată din 4 obiecte: *Emițătorul*, *Receptorul*, canalul cuantic și canalul clasic. *Emițătorul* va transmite qbiții prin canalul cuantic iar *Receptorul* va extrage acești qbiți din canalul cuantic. La finalul transmisiei cuantice, *Emițătorul* și *Receptorul* vor comunica pe canalul clasic și vor executa etapele *Bases reconciliation*, *Secret key reconciliation* și *Privacy amplification*.

7.2.2 Implementarea hardware

Schema bloc a programului BB84-Ideal este prezentată în figura 7.2.

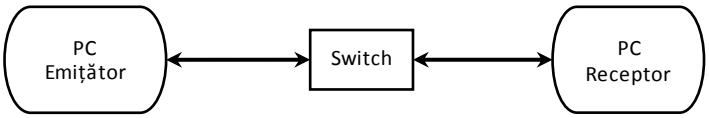

Figura 7.2. Schema bloc a programului de simulare BB84-Ideal

Echipamentele utilizate pentru implementarea programului de simulare BB84-Ideal sunt:

- 2 calculatoare;
- 1 switch.

Echipamentele sunt conectate între ele prin intermediul unui switch. Fiecare calculator va avea IP static, pentru a putea comunica prin intermediul switch-ului și va rula un program specific emițătorului respectiv receptorului.

7.2.3 Implementarea protocolului

Schema funcțională a programului de simulare BB84-Ideal este prezentată în figura 7.3.

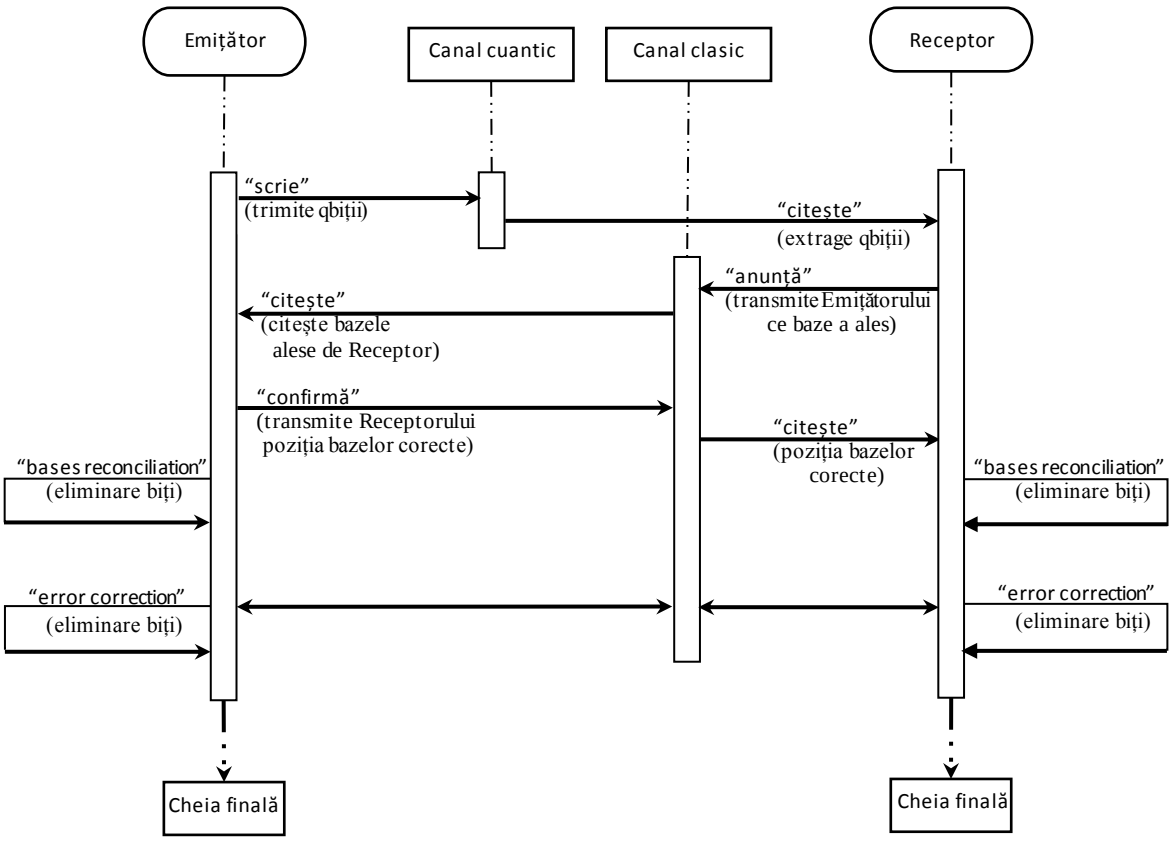

Figura 7.3. Schema funcțională a programului de simulare BB84-Ideal

7.2.4 Rezultate

În urma rulării programului de simulare BB84-ideal de 10 ori, obținem următoarele rezultate, tabelul 7.2, pentru o cheie inițială care are dimensiunea de 320 biți:

Cheia inițială	Cheia finală	QBER %
320	162	49.4
320	161	49.7
320	152	52.5
320	172	46.3
320	158	50.6
320	153	52.2
320	165	48.4
320	163	49.1
320	150	53.1
320	160	50.0

Tabelul 7.2. Simulare BB84 în condiții ideale

Analizând datele obținute putem constata că quantum bit error rate – QBER din *cheia finală* este aproximativ 50%.

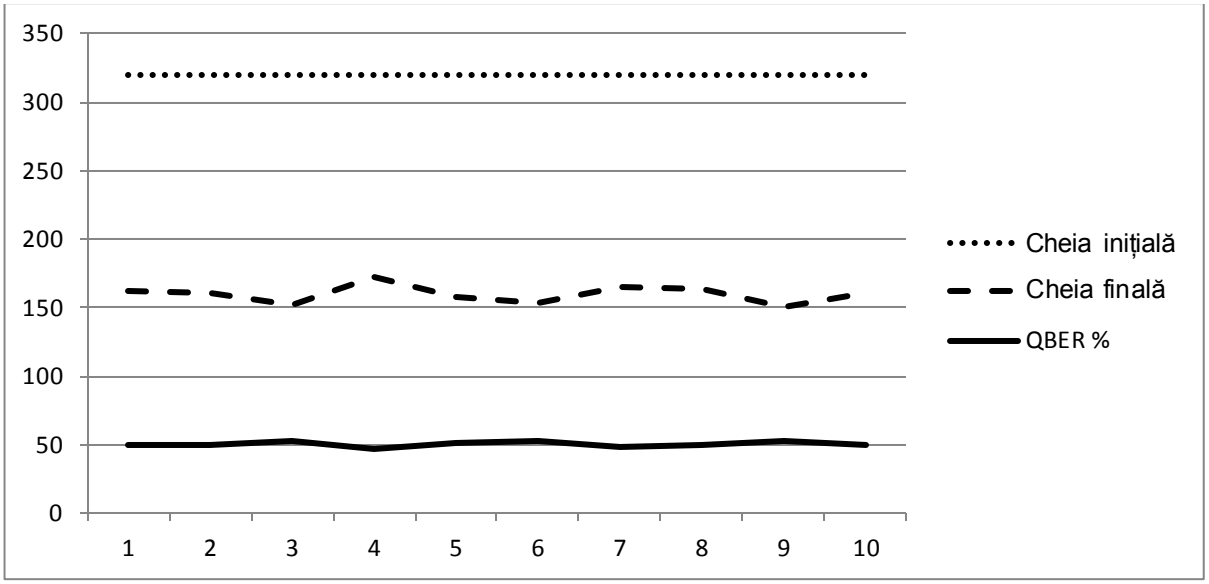

Figura 7.4. Detalii simulare BB84 în condiții ideale

7.2.5 Detalii de funcționare

Programul *BB84-ideal* este format din două programe, PCEmitator și PCReceptor, care rulează pe calculatoare diferite și simulează un sistem de distribuire a cheilor cuantice BB84, în condiții ideale și fără prezența inamicului. Cele două programe comunică între ele prin intermediul unui switch care va simula canalul clasic și cel cuantic.

*PC - Emițător * BB84 Algorithm* pregătește secvența inițială de biți pe care dorește să o transmită către *receptor* și secvența de baze de polarizare pe care le va aplica fiecărui foton în parte pentru a reprezenta biții, conform tabelului 7.1.

Qbiții pe care *emițătorul* îi va transmite către *receptor* sunt reprezentați în fereastra *Sender's Qbits*, figura 7.5.

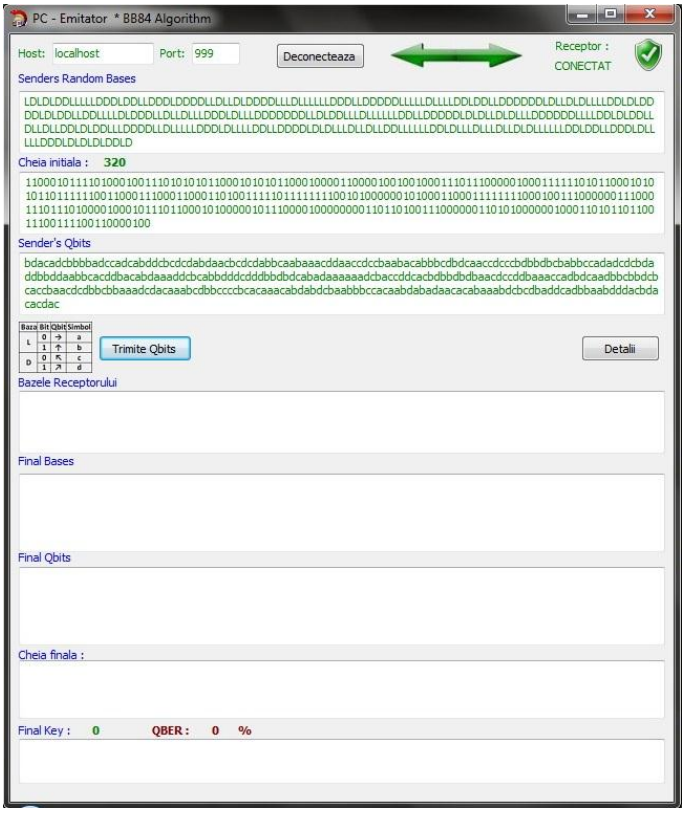

Figura 7.5. Emițătorul - Bazele de polarizare, biții și qbiții pregătiți de emițător

*PC - Receptor * BB84 Algorithm* pregătește la rândul său bazele de polarizare cu care va citi qbiții recepționați de la *emițător*, conform tabelului 7.1.

În câmpurile *Host* și *Port* se introduc IP-urile celor două stații de lucru și porturile pe care acestea vor comunica și se apasă butoanele conectează ale celor două programe pentru a realiza legăturile dintre *emițător* și *receptor*.

Prin apăsarea butonului *Trimite Qbits*, *emițătorul* va transmite către *receptor* qbiții pregătiți de către acesta.

În figura 7.6 este prezentat *receptorul*, în ipostaza conectat la *emițător*, în câmpul *Receivers Random Bases* se pot observa bazele de polarizare alese de către acesta iar în câmpul *Received Qbits* avem qbiții citiți de către *receptor* în funcție de bazele de polarizare alese, conform tabelului 7.1.

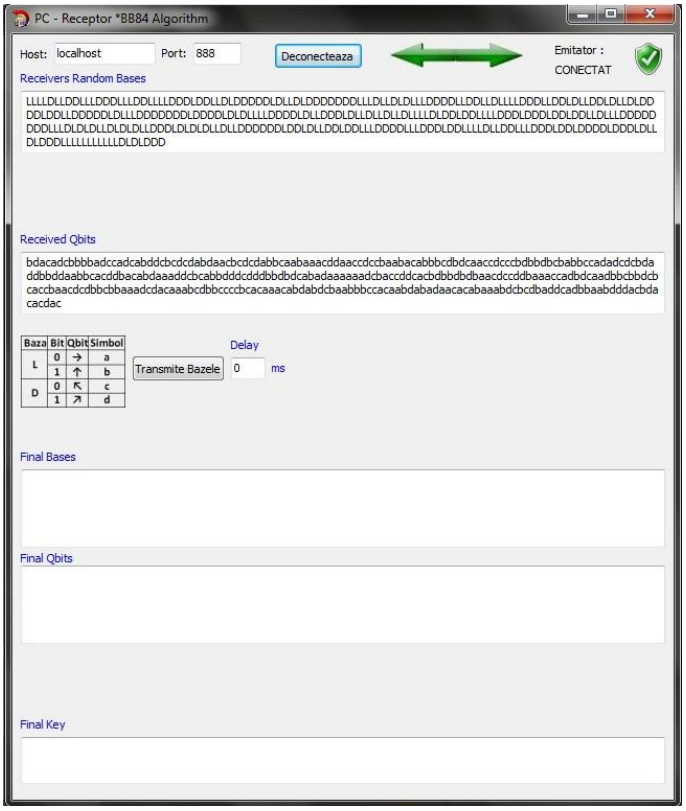

Figura 7.6. Receptorul - Bazele de polarizare și qbiții receptorului

Următoarea etapă a algoritmului BB84 se numește etapa Bases Reconciliation și constă în comunicarea, pe canalul clasic dintre *emițător* și *receptor*, a bazelor de polarizare alese pentru codificarea respectiv decodificarea biților, urmată de eliminarea biților pentru care nu au fost alese aceleași baze de polarizare. Aceasta se realizează prin butonul *Transmite Bazele*.

După etapa de eliminare a biților pentru care nu au fost alese aceleași baze de polarizare – fereastra *Final Bases* din *PC Emițător* și *PC Receptor*, obținem qbiții finali – fereastra *Final Qbits*, iar prin aplicarea convenției stabilite în tabelul 7.1, obținem cheia finală.

În final transmisia este realizată iar schimbul de chei cuantice finalizat, așa cum se poate vedea din figura 7.7. – fereastra *Final Key*.

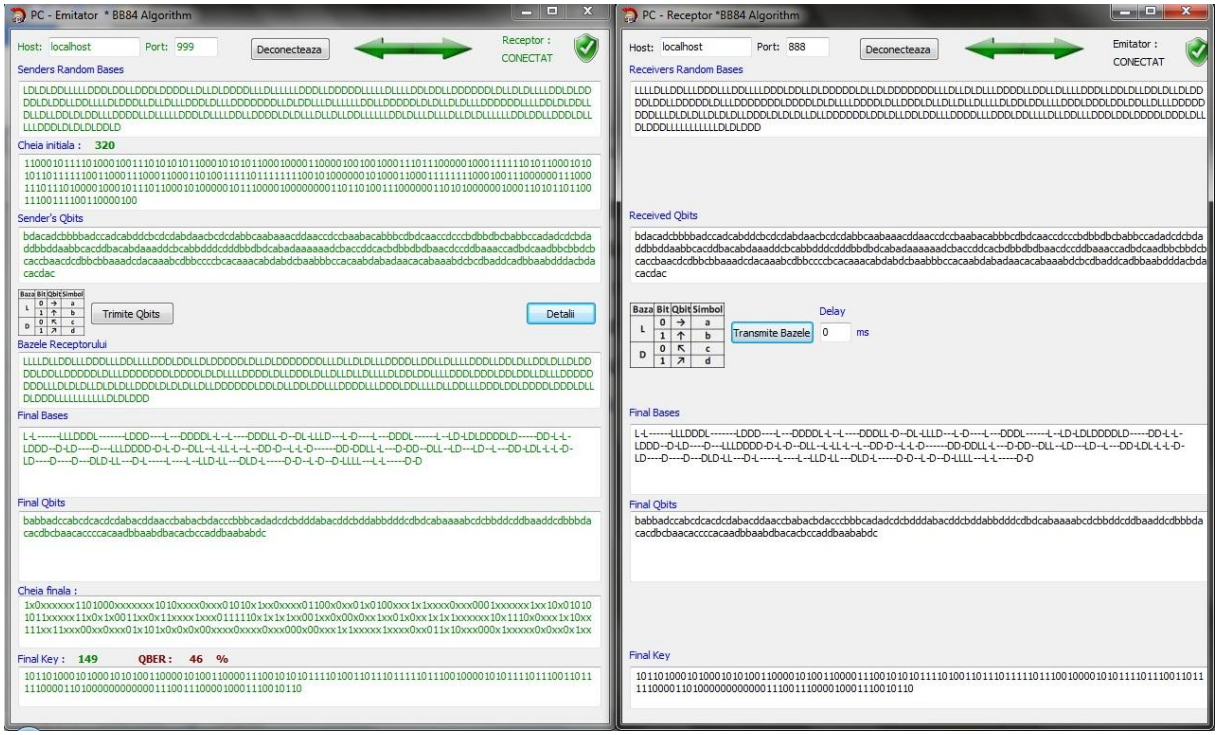

Figura 7.7. Emițătorul și Receptorul – obținerea cheii finale unice, sigure și secrete

Prin butonul *Detalii* din cadru *PC Emițător*, se pot vedea detaliile privind eliminarea sau păstrarea biților din cheia primară – figura 7.8.

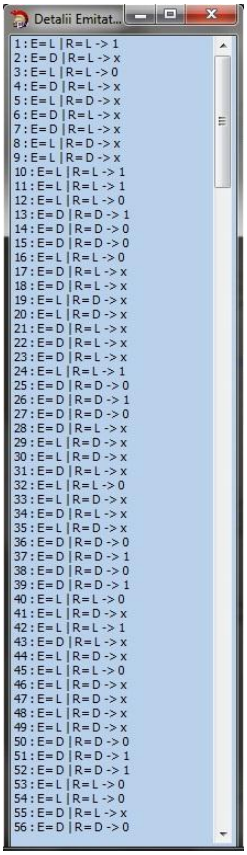

Figura 7.8. Detalii

7.3 Simularea sistemului BB84 în condiții ideale cu inamic

7.3.1 Implementarea software

Pentru implementarea programului de simulare BB84 am folosit limbajul C++. *Emițătorul* și *Receptorul* vor comunica prin canalul cuantic și cel clasic, cu sau fără prezența *Inamicului* iar legătura între ei se va face prin intermediul unui switch.

Această aplicație software este formată din 5 obiecte: *Emițătorul*, *Receptorul*, *Inamicul*, canalul cuantic și canalul clasic. *Emițătorul* va transmite qbiții prin canalul cuantic către *Receptor*. *Inamicul*, va întrerupe canalul cuantic, va intercepta qbiții transmiși de către *Emițător*, pe care îi va citi în conformitate cu bazele alese de el și va transmite către *Receptor* alți qbiți în conformitate cu bazele alese de către *Inamic*. *Receptorul* va extrage acești qbiți din canalul cuantic iar la finalul transmisiei cuantice, *Emițătorul* și *Receptorul* vor comunica pe canalul clasic și vor executa etapele *Bases reconciliation*, *Secret key reconciliation* și *Privacy amplification*.

Metoda de atac folosită de către *inamic* în programul de simulare BB84 este metoda Intercept-Resend. Atacul de tip Intercept-Resend [52], este cel mai întâlnit tip de atac folosit asupra sistemelor de distribuire a cheilor cuantice. *Inamicul*, întrerupe canalul cuantic – figura 7.9, măsoară fiecare qbit recepționat de la *emițător* în una dintre cele două baze de polarizare, alese aleatoriu de către el și transmite către *receptor* alți qbiți polarizați în baze de polarizare corespunzătoare cu rezultatele obținute de către el, fără a lăsa urme ale atacului [28].

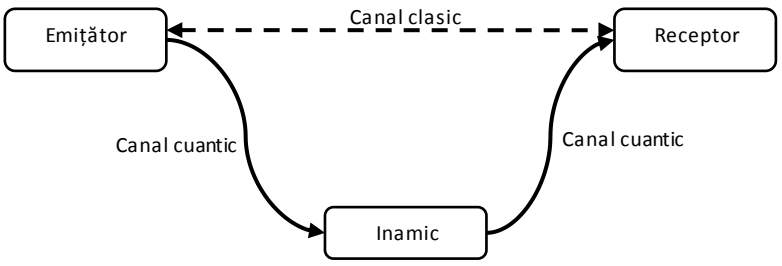

Figura 7.9. Atac de tip Intercept-Resend

După etapele *Secret key reconciliation* și *Privacy amplification*, *emițătorul* și *receptorul* dețin aceeași *cheie finală*, unică și secretă care va fi folosită împreună cu algoritmul one-time pad pentru cripta informația pe care dorim să o transmitem către *receptor*.

7.3.2 Implementarea hardware

Schema bloc a programului de simulare BB84 este prezentată în figura 7.1. Echipamentele utilizate pentru implementarea programului de simulare BB84 sunt:

- 3 calculatoare;
- 1 switch.

Echipamentele sunt dispuse în aceeași încăpere și sunt conectate între ele prin intermediul unui switch. Fiecare calculator va avea IP static, pentru a putea comunica prin intermediul switch-ului și va rula un program specific emițătorului, receptorului și respectiv inamicului.

7.3.3 Implementarea protocolului

Schema funcțională programului de simulare a sistemului cuantic de comunicații BB84 cu prezența *inamicului* este prezentată în figura 7.10.

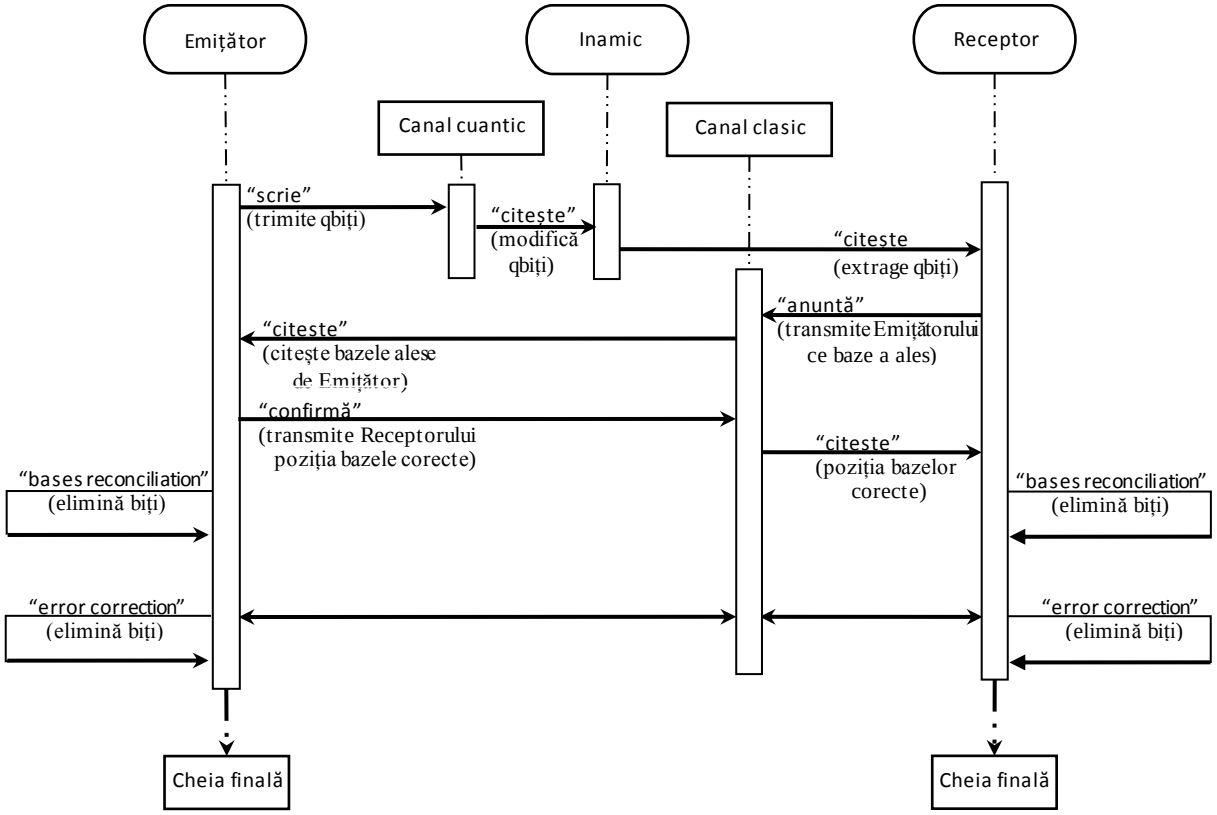

Figura 7.10. Schema funcțională a simulării protocolului BB84

7.3.4 Rezultate

În urma rulării programului de simulare BB84 cu inamic de 10 ori, obținem următoarele rezultate, tabelul 7.3, pentru o cheie inițială care are dimensiunea de 320 biți:

Cheia inițială	Cheia brută	Cheia finală	QBER %
320	149	110	65.6
320	156	114	64.4
320	172	129	59.7
320	164	131	59.1
320	167	126	60.6
320	144	105	67.2
320	159	105	67.2
320	133	108	66.3
320	162	117	63.4
320	171	123	61.6

Tabelul 7.3. Simulare BB84 în condiții ideale cu inamic

Analizând datele obținute putem constata că quantum bit error rate – QBER din *cheia finală* este aproximativ 64 % – figura 7.11.

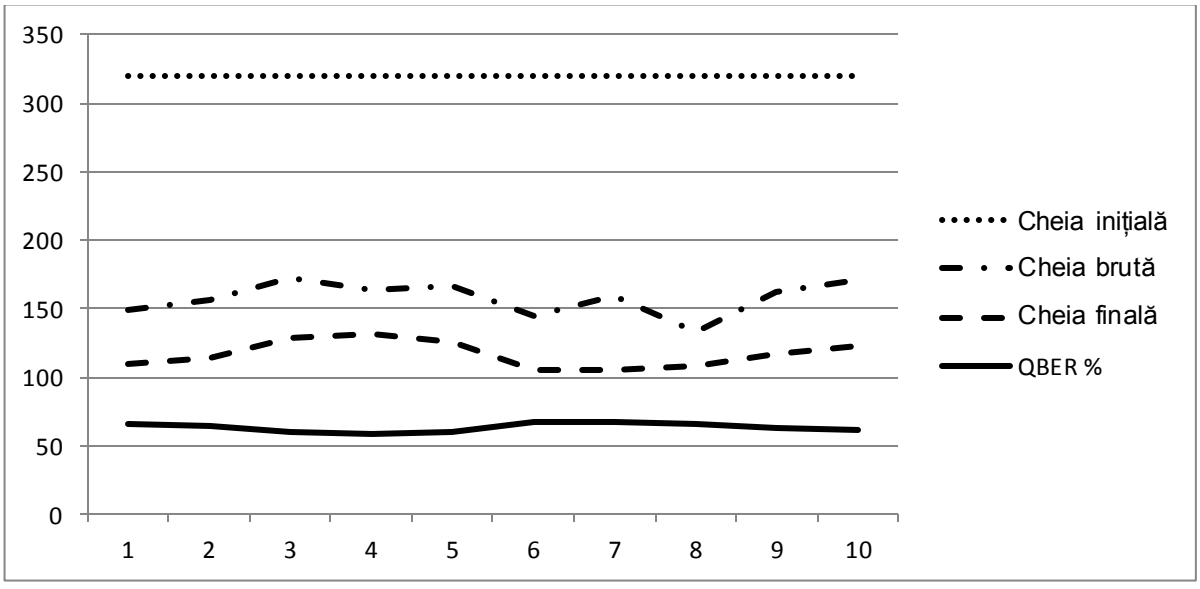

Figura 7.11. Detalii simulare BB84 în condiții ideale cu inamic

7.3.5 Detalii de funcționare

Prezența *inamicului* în sistemul de distribuire a cheilor cuantice BB84 va modifica qbiții din cheia recepționată de către *receptor* în funcție de bazele de polarizare alese de către *inamic*.

PC - Emițător BB84 pregătește secvența inițială de biți pe care dorește să o transmită către *receptor* și secvența de baze de polarizare pe care le va aplica fiecărui foton în parte pentru a reprezenta biții, conform tabelului 7.1.

Qbiții pe care *emițătorul* îi va transmite către *receptor* sunt reprezentați în fereastra *Sender's Qbits*, biții pe care dorește să-i codifice sunt reprezentați în fereastra *Random Bits*, iar bazele de polarizare sunt reprezentate în fereastra *Senders Random Bases* din figura 7.12.

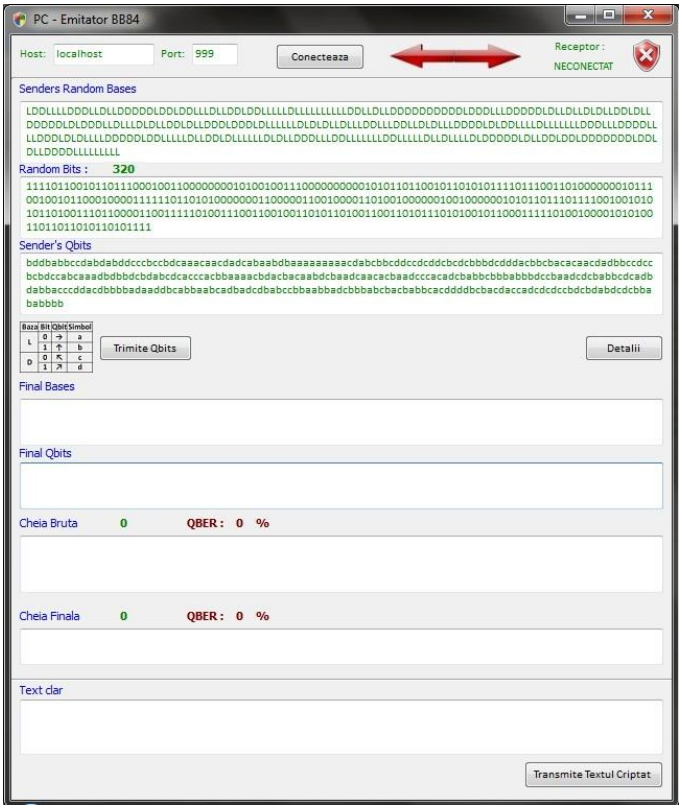

Figura 7.12. Emițătorul - Bazele de polarizare, biții și qbiții pregătiți de emițător

PC – Inamic BB84 alege ce baze de polarizare să folosească pentru interpretarea qbiților pe care îi va intercepta, întrerupe canalul cuantic dintre *emițător* și *receptor*, măsoară fiecare qbit recepționat de la *emițător* în una dintre cele două baze de polarizare alese și transmite către *receptor* alți qbiți polarizați în baze de polarizare corespunzătoare cu rezultatele obținute de către el.

În figura 7.13 este prezentat *inamicul*, se pot observa qbiții interceptați – fereastra *Qbiții Emițătorului* și în funcție de filtrele de polarizare aplicate, *inamicul* obține o altă secvență de qbiți – fereastra *Qbiții interceptați de Inamic*, pe care îi va transmite către *receptor* – butonul *Trimite Qbits*.

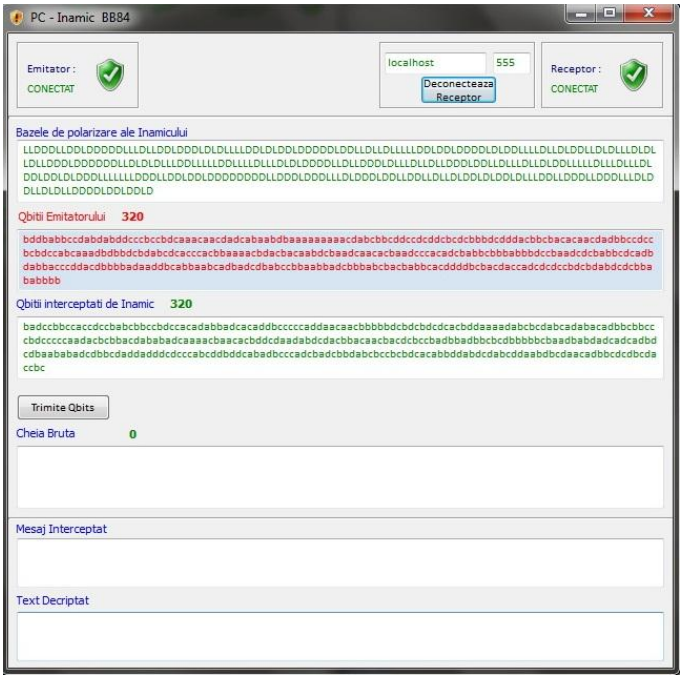

Figura 7.13. Inamicul - Bazele de polarizare și qbiții interceptați

Următoarea etapă a algoritmului BB84 se numește etapa *bases reconciliation* și constă în comunicarea, pe canalul clasic dintre *emițător* și *receptor*, a bazelor de polarizare alese pentru codificarea respectiv decodificarea biților, urmată de eliminarea biților pentru care nu au fost alese aceleași baze de polarizare. Aceasta se realizează prin butonul *Transmite Bazele*.

După etapa de eliminare a biților pentru care nu au fost alese aceleași baze de polarizare – fereastra *Final Bases* din *PC Emițător* și *PC Receptor*, obținem qbiții finali – fereastra *Final Qbits*, iar prin aplicarea convenției stabilite în tabelul 7.1, obținem *cheia brută*, figura 7.14.

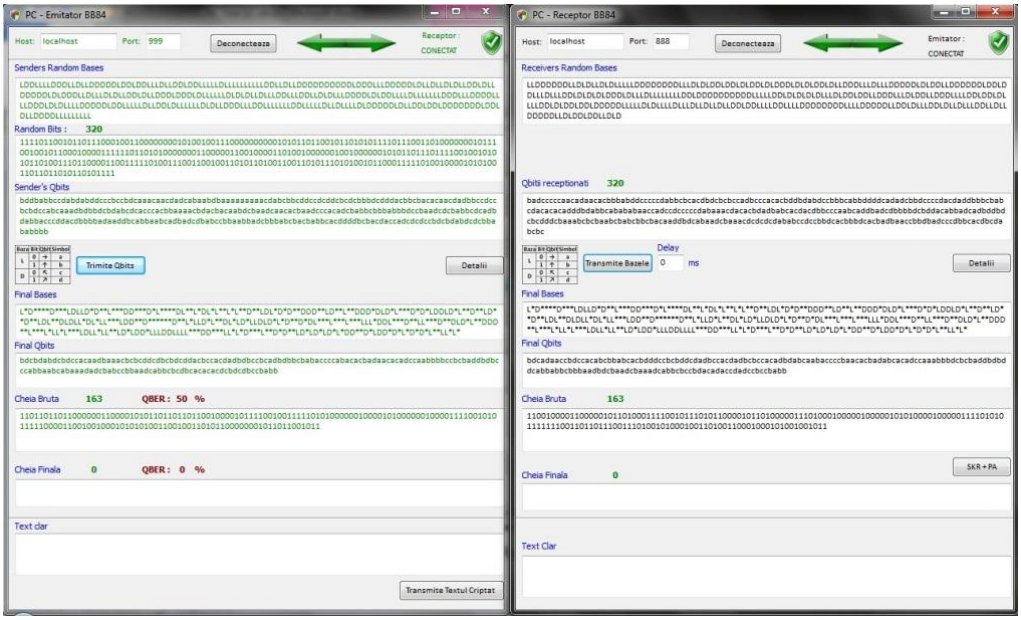

Figura 7.14. Emițătorul și Receptorul – obținerea cheii brute

Datorită faptului că *inamicul* a fost prezent și a interceptat și modificat transmisia, cheia obținută de către *emițător* și *receptor* nu este identică.

Următoarele etape ale algoritmului BB84 se numesc *Secret key reconciliation* și *Privacy amplification*, care se vor realiza prin acționarea butonului – *SKR + PA*.

Etapa *Secret key reconciliation* realizează o căutare binară, interactivă a erorilor. *Emițătorul* și *receptorul* împart secvența de biți rămasă (cheia brută) în blocuri de biți și vor compara paritatea fiecărui bloc. În cazul în care paritatea unui bloc de biți diferă, *emițătorul* și *receptorul* vor împărți blocul respectiv în blocuri mai mici și vor compara paritatea lor.

Acest proces va fi repetat până când bitul care diferă va fi descoperit și eliminat. Comunicațiile din această etapă se vor realiza pe un canal public nesecurizat la sfârșitul cărei se va obține cheia secretă.

Având în vedere faptul că, în etapa *Secret key reconciliation*, comunicația s-a realizat pe un canal nesecurizat, există posibilitatea ca *inamicul* să dețină informații sensibile despre cheia secretă. Pentru a stabili o cheie perfect sigură, *emițătorului* și *receptorului* trebuie să mai realizeze o etapă : *Privacy amplification.* Această etapă constă într-o permutare a biților din cheia secretă și eliminarea unui subset de biți, care va fi realizată de către *emițător* și *receptor*.

La finalul acestei etape avem certitudinea că *emițătorul* și *receptorul*, dețin aceeași cheie unică, secretă care nu este cunoscută de *inamic* cum se poate vedea din figura 7.15.

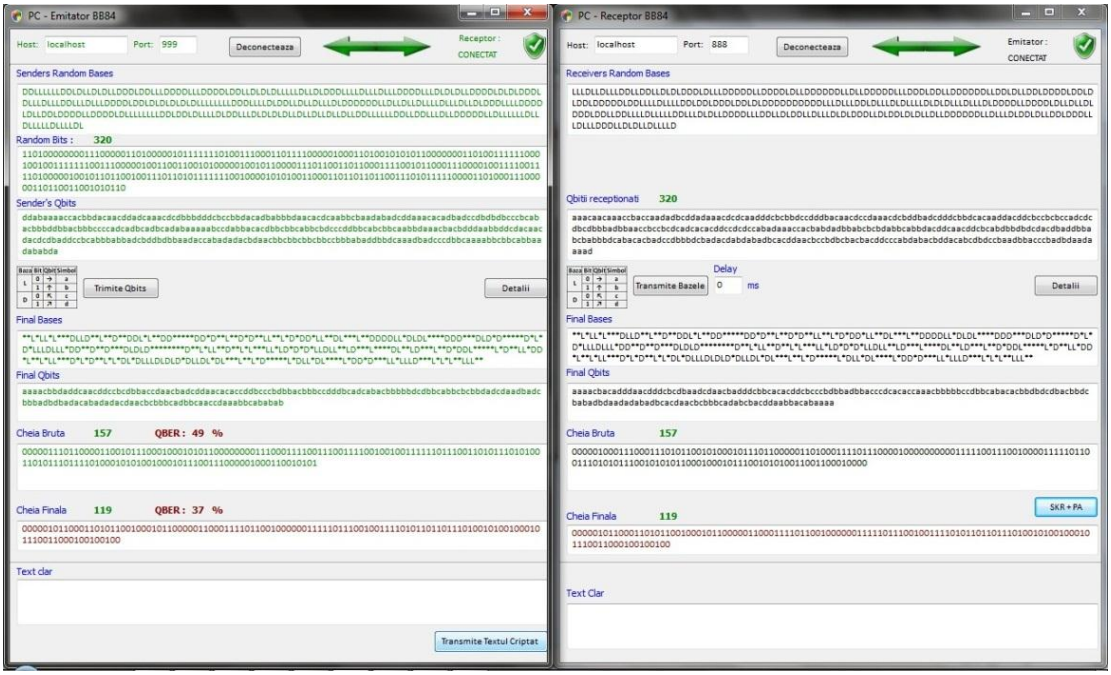

Figura 7.15. Emițătorul și Receptorul – obținerea cheii finale

În acest moment *emițătorul* și *receptorul* dețin o cheie finală unică, sigură și secretă care poate fi folosită împreună cu algoritmul one-time pad. Algoritmul one-time pad este implementat în programul de simulare și în figura 7.16 se poate observa textul interceptat și decriptat de către *inamic* și cel recepționat și decriptat de către *receptor*.

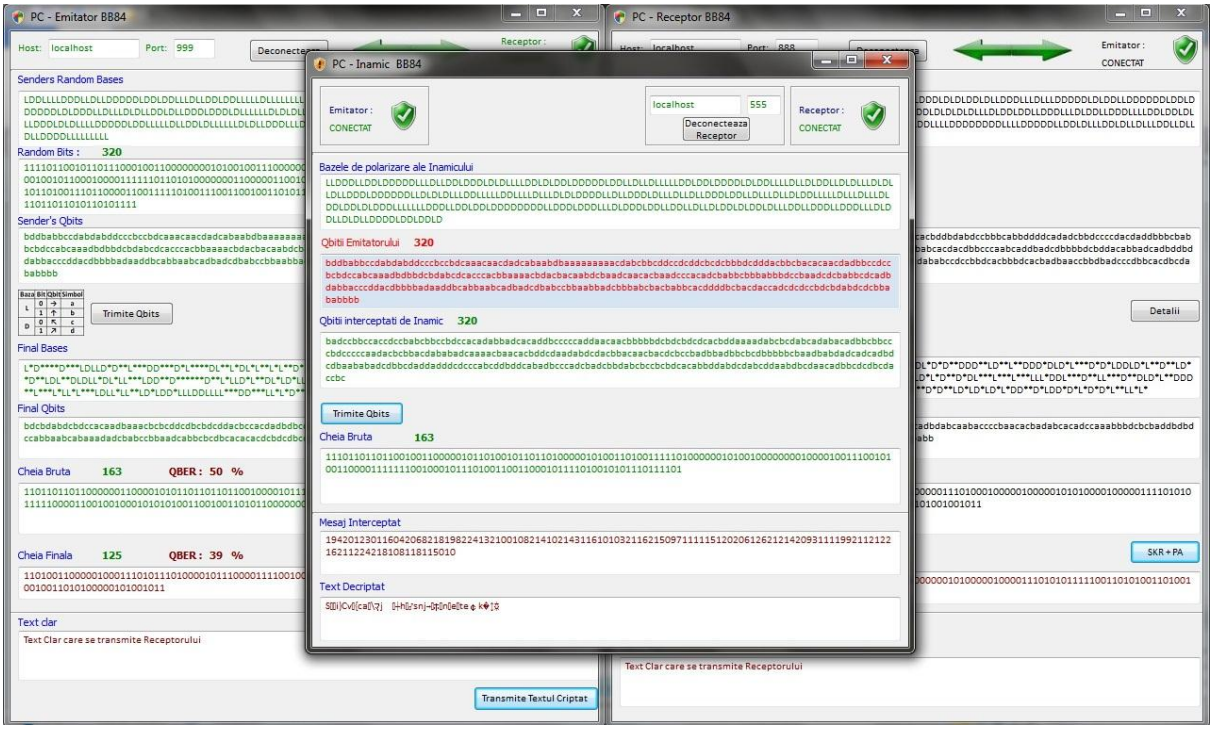

Figura 7.16. Emițătorul, Receptorul și Inamicul

În concluzie, putem spune că în urma schimbului de chei cuantice dintre *emițător* și *receptor* a fost obținută o *cheie finală* unică și secretă cu care a fost criptat un text, cu algoritmul *one-time pad*, care chiar dacă a fost interceptat de către *inamic*, acesta nu a putut fi decriptat.

7.4 Simularea sistemului BB84 cu metoda QBTT

7.4.1 Implementarea software

Programul BB84 QBTT a fost scris în limbajul C++ OOP și simulează un sistem de distribuire a cheilor cuantice de tip BB84, în condiții ideale cu prezența *inamicului*, care folosește pentru detectarea inamicului metoda QBTT.

Implementarea software a acestui program de simulare este aproximativ identică cu cea a programului de simulare BB84 cu excepția că detectarea *inamicului* se face prin metoda QBTT.

În primă fază *emițătorul* și *receptorul* își vor sincroniza ceasurile interne ale dispozitivelor de lucru, în cazul nostru al calculatoarelor.

Emițătorul, pentru fiecare qbit transmis către *receptor* va transmite, pe canalul clasic timestamp-ul momentului emisiei time-stamp_e. La rândul său, *receptorul*, pentru fiecare qbit recepționat, citește timestamp-ul transmis de către *emițător* și generează timestamp-ul momentului recepției time-stamp_r, calculând întârzierea ΔT= time-stamp_r - time-stamp_e.

Dacă *inamicul* interceptează qbiții transmiși de către *emițător*, *receptorul* va sesiza acest lucru prin creșterea timpului parcurs de qbit și va opri transmisia, prin calcularea întârzierii ΔT = time-stamp_r - time-stamp_e.

7.4.2 Implementarea hardware

Schema bloc a programului de simulare BB84 este prezentată în figura 7.1. Echipamentele utilizate pentru implementarea programului de simulare BB84 sunt:

- 3 calculatoare;
- 1 switch.

Echipamentele sunt dispuse în aceeași încăpere și sunt conectate între ele prin intermediul unui switch. Fiecare calculator va avea IP static, pentru a putea

comunica prin intermediul switch-ului și va rula un program specific emițătorului, receptorului și respectiv inamicului.

7.4.3 Implementarea protocolului

Schema funcțională a programului de simulare a sistemului cuantic de comunicații BB84 cu metoda QBTT de detectare a *inamicului* este prezentată în figura 7.17.

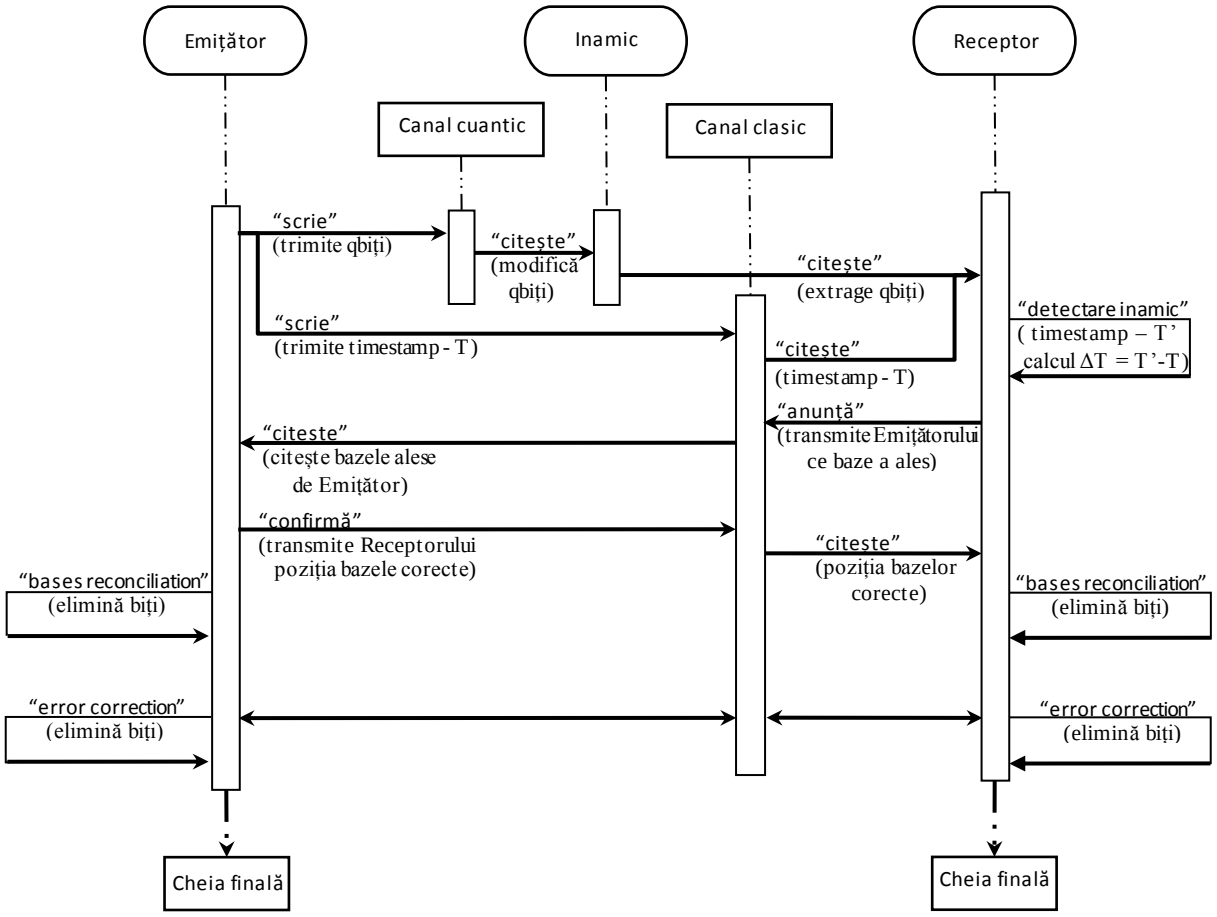

Figura 7.17. Schema funcțională a programului de simulare BB84 QBTT

7.4.4 Rezultate

În urma rulării programului de simulare BB84 cu metoda QBTT de 10 ori, obținem următoarele rezultate, tabelul 7.4, pentru o cheie inițială care are dimensiunea de 320 biți:

Cheia inițială	Cheia brută	Cheia finală	QBER %
320	148	148	46.2
320	150	150	46.8
320	167	167	52.1
320	157	157	49
320	159	159	49.6
320	152	152	47.5
320	168	168	52.5
320	157	157	49
320	158	158	49.3
320	162	162	50.6

Tabelul 7.4. Simulare BB84 în condiții ideale cu metoda QBTT

Analizând datele obținute putem constata că procentul de erori din cheia finală, quantum bit error rate – QBER este aproximativ 50%, ca în cazul algoritmului BB84 în condiții ideale fără inamic, iar *cheia brută* este egală cu *cheia finală* - figura 7.18.

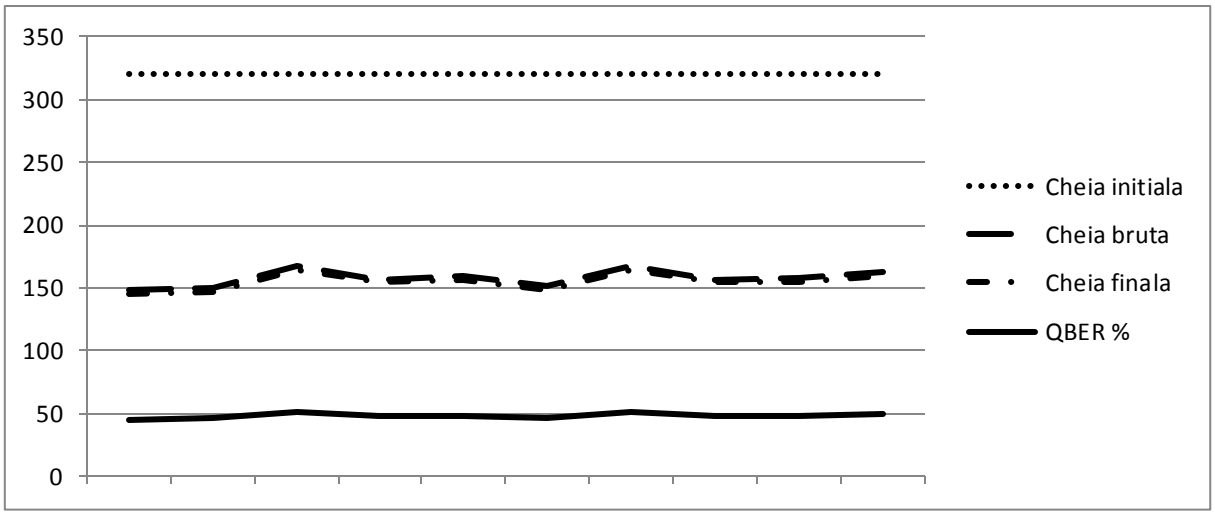

Figura 7.18. Detalii simulare BB84 în condiții ideale cu inamic

7.4.5 Detalii de funcționare

PC - Emițător BB84 QBTT pregătește secvența inițială de biți pe care dorește să o transmită către *receptor* și secvența de baze de polarizare pe care le va aplica fiecărui foton în parte pentru a reprezenta biții, conform tabelului 7.1.

Qbiții pe care *emițătorul* îi va transmite către *receptor* sunt reprezentați în fereastra *Sender's Qbits* după finalizarea transmisiei cuantice, biții pe care dorește

să-i codifice sunt reprezentați în fereastra *Random Bits*, iar bazele de polarizare sunt reprezentate în fereastra *Senders Random Bases* din figura 7.19.

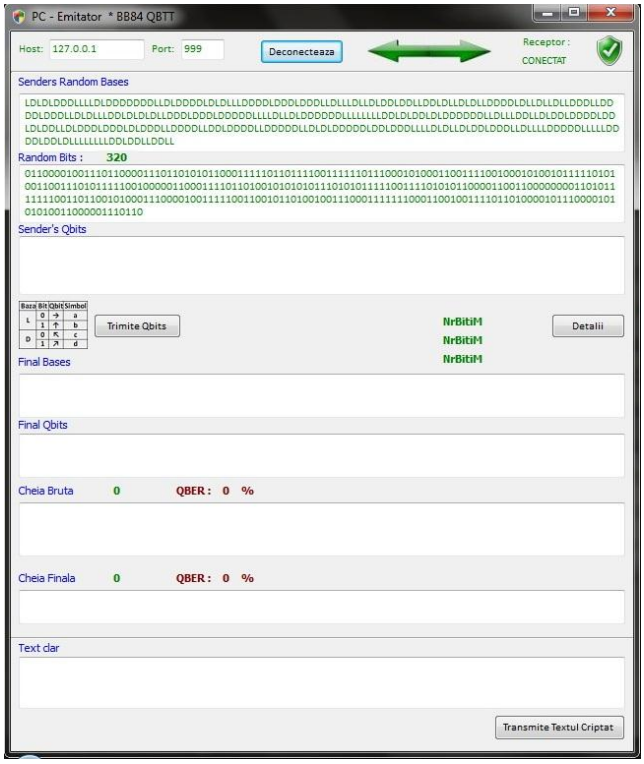

Figura 7.19. PC - Emițătorul BB84 QBTT

Programul PC – Inamic BB84 QBTT, va intercepta transmisia cuantică dintre *emițător* și *receptor* doar în momentele în care căsuța Interceptează Qbiții este bifată, figura 7.20, mărind durata parcursă de către qbit de la *emițător* la *receptor*.

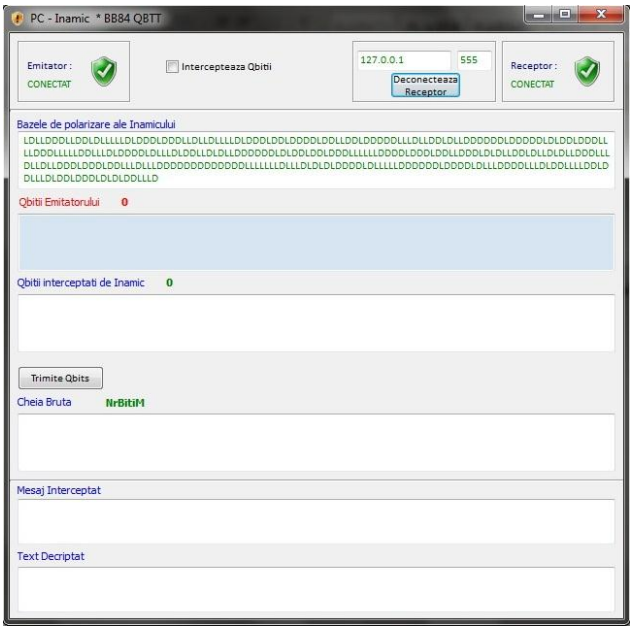

Figura 7.20. PC - Inamic BB84 QBTT

În momentul în care *inamicul* interceptează qbiții transmiși de către emițător, *receptorul* va sesiza acest lucru prin creșterea timpului parcurs de qbit și va opri transmisia, figura 7.21.

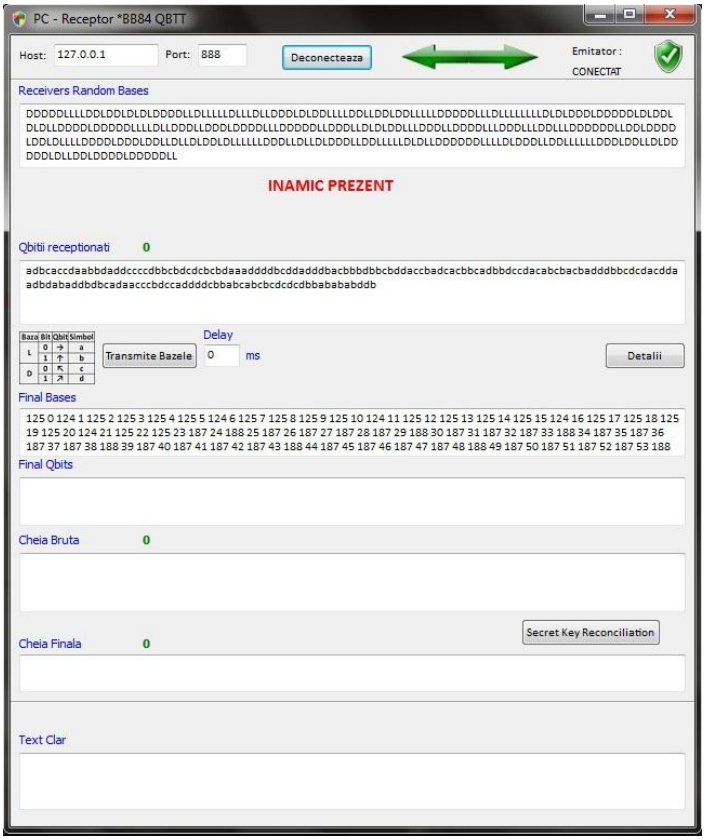

Figura 7.21. PC - Receptor BB84 QBTT

7.5 Simularea protocolului BSPA

7.5.1 Implementarea software

Pentru implementarea programului de simulare BSPA am folosit limbajul C++. *Emițătorul* și *Receptorul* vor comunica prin canalul cuantic și cel clasic, cu sau fără prezența *Inamicului* iar legătura între ei se va face prin intermediul unui switch.

Această aplicație software este formată din 5 obiecte: *Emițătorul*, *Receptorul*, *Inamicul*, canalul cuantic și canalul clasic.

În primă fază *emițătorul* și *receptorul* își vor sincroniza ceasurile interne ale dispozitivelor de lucru, în cazul nostru al calculatoarelor.

Prin protocolul cuantic de comunicații BSPA, *emițătorul* și *receptorul* vor stabili în comun, în funcție de biții din *cheia finală* obținută cu ajutorul oricărui sistem de

distribuire a cheilor cuantice, care pereche, dintre cele trei baze de polarizare, *liniar-diagonal*, *liniar-circular* sau *diagonal-circular* vor fi folosite pentru polarizarea fotonilor.

În funcție de biții rămași din *cheia finală*, *emițătorul* și *receptorul*, vor ști exact ce baze de polarizare să aplice pentru fiecare foton pe care urmează să-l transmită respectiv să-l recepționeze.

Emițătorul, pentru fiecare qbit transmis către *receptor* va transmite, pe canalul clasic timestamp-ul momentului emisiei time-stamp_e. La rândul său, *receptorul*, pentru fiecare qbit recepționat, citește timestamp-ul transmis de către *emițător* și generează timestamp-ul momentului recepției time-stamp_r, calculând întârzierea ΔT= time-stamp_r - time-stamp_e.

Dacă *inamicul* interceptează qbiții transmiși de către *emițător*, *receptorul* va sesiza acest lucru prin creșterea timpului parcurs de qbit și va opri transmisia, prin calcularea întârzierii ΔT = time-stamp_r - time-stamp_e.

7.5.2 Implementarea hardware

Schema bloc a programului de simulare BSPA este prezentată în figura 7.1. Echipamentele utilizate pentru implementarea programului de simulare BSPA sunt:

- 3 calculatoare;
- 1 switch.

Echipamentele sunt dispuse în aceeași încăpere și sunt conectate între ele prin intermediul unui switch. Fiecare calculator va avea IP static, pentru a putea comunica prin intermediul switch-ului și va rula un program specific emițătorului, receptorului și respectiv inamicului.

7.5.3 Implementarea protocolului

Schema funcțională a programului de simulare a protocolului BSPA este prezentată în figura 7.22.

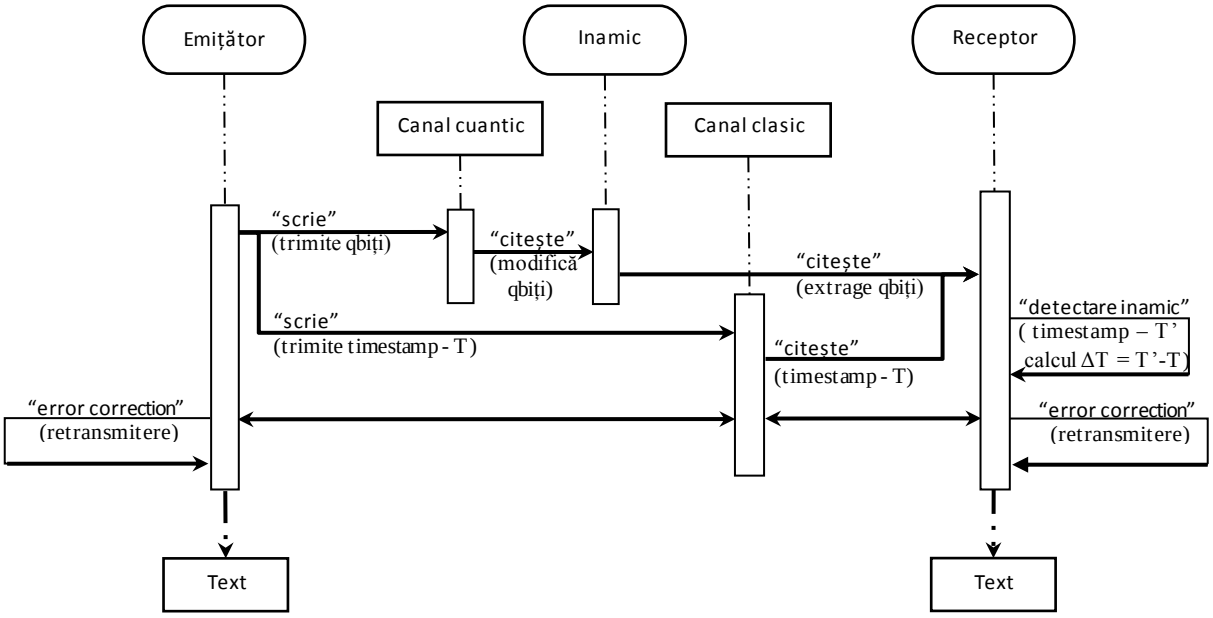

Figura 7.22. Schema funcțională a programului de simulare BSPA

7.5.4 Rezultate

În urma rulării programului de simulare BSPA de 10 ori, obținem următoarele rezultate, tabelul 7.5, pentru o cheie inițială care are dimensiunea de 320 biți:

Cheia inițială	Cheia finală	QBER %
320	320	0.0
320	317	0.9
320	320	0.0
320	319	0.3
320	320	0.0
320	317	0.9
320	320	0.0
320	319	0.3
320	320	0.0
320	319	0.3

Tabelul 7.5. Simulare BSPA

Analizând datele obținute putem constata că, *cheia finală* are aproximativ aceeași lungime cu *cheia inițială* iar procentul de erori din *cheia finală*, quantum bit error rate – QBER este de 0.27 % - figura 7.23.

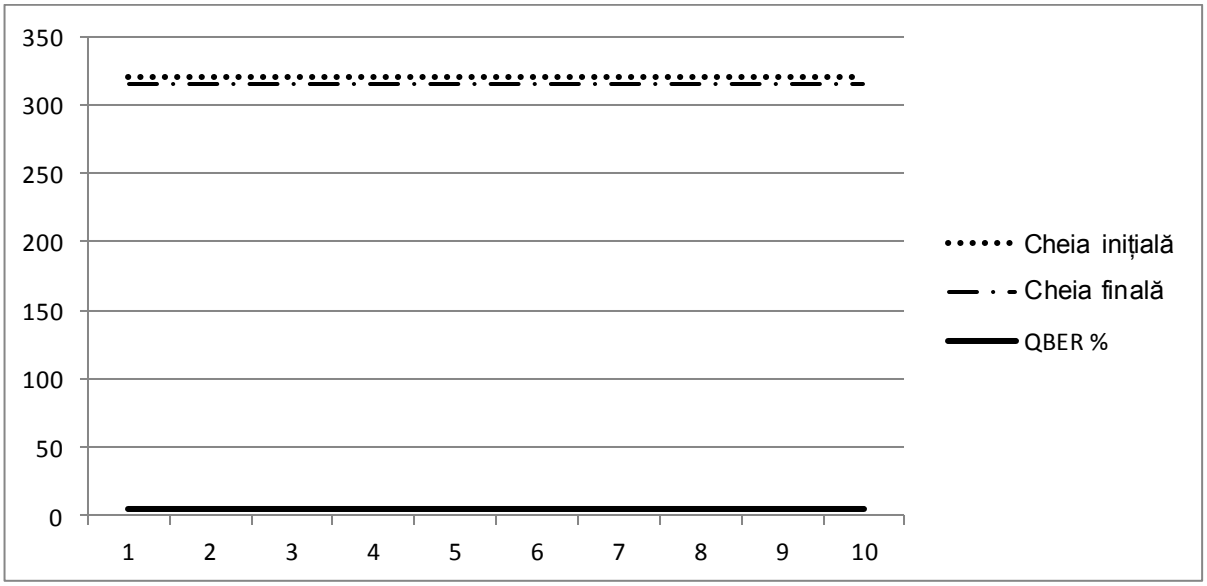

Figura 7.23. Detalii simulare BSPA

7.5.5 Detalii de funcționare

PC – Emițător BSPA și PC – Receptor BSPA dețin aceeași cheie finală și în funcție de aceasta vor stabili în comun ce baze de polarizare (linear-diagonal, linear-circular sau diagonal-circular) vor folosi pentru polarizarea biților. Tot în funcție de cheia finală ei vor stabili ce baze de polarizare vor aplica fiecăruli qbit în parte, pentru polarizare respective pentru citirea qbiților.

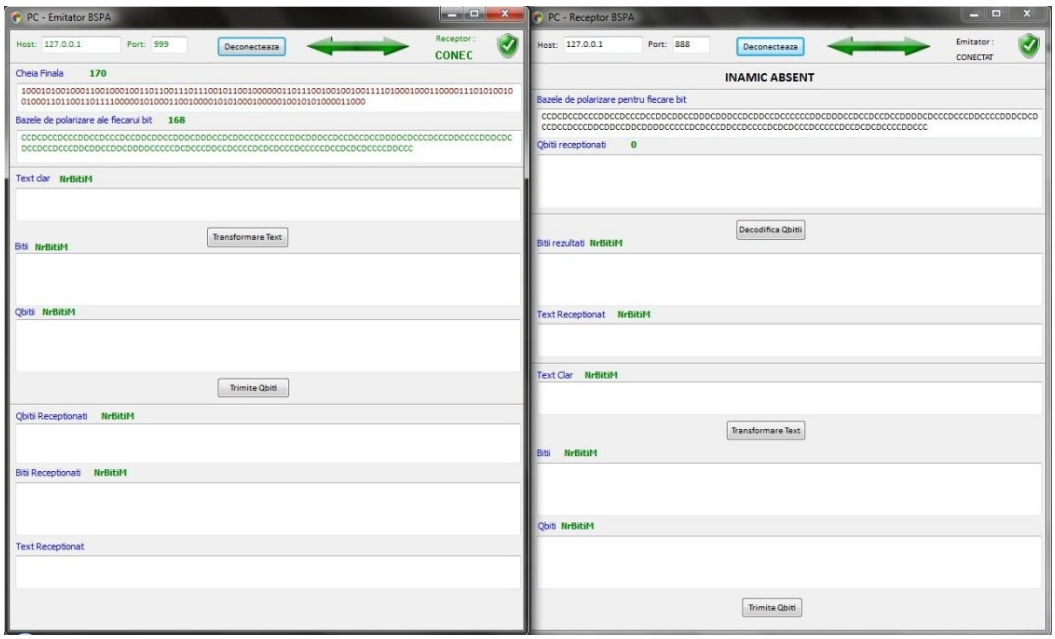

Figura 7.24. Emițătorul și receptorul în protocolul BSPA

În câmpul *Text Clar* din cadrul PC-Emitator BSPA se va introduce textul care se dorește să fie trimis către Receptor. Butonul *Transformare Text* transformă fiecare caracter ascii în binar iar biții sunt prezentați în câmpul *Bitii*. Fiecare bit va fi transformat în qbit și va fi afișat în câmpul *Qbiții*.

Prin acționarea butonului *Trimite Qbiții*, Emițătorul va transmite, pe canalul cuantic, catre Receptor qbiții.

Receptorul, prin acționarea butonului *Decodifică Qbiții* va decodifica qbiții recepționați în biți, campul *Biții rezultați* și apoi în text clar, câmpul *Text Recepționat*.

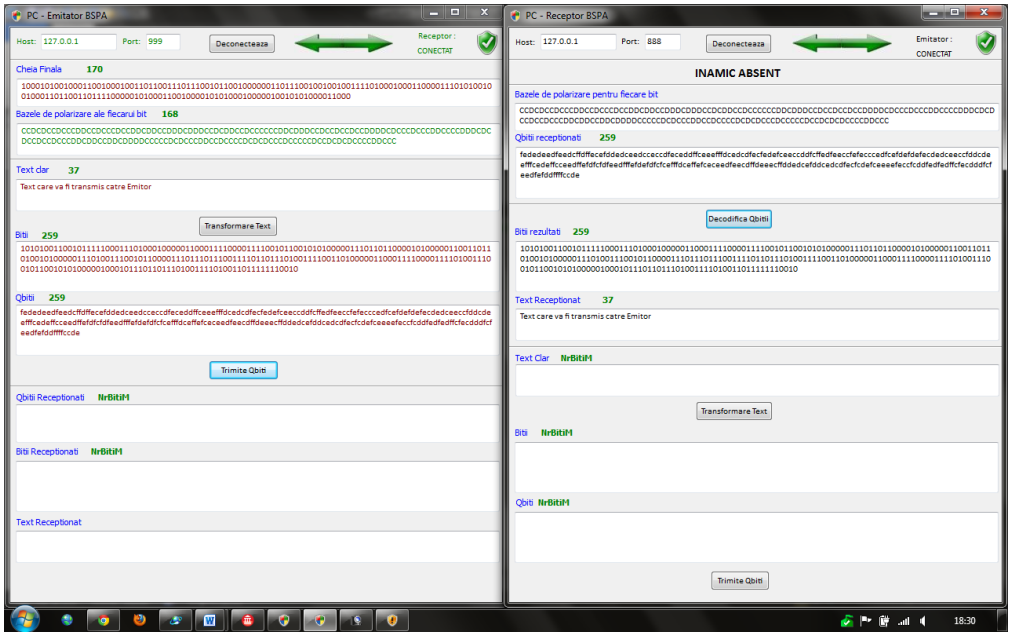

Figura 7.25. Detalii de funcționare în protocolul BSPA

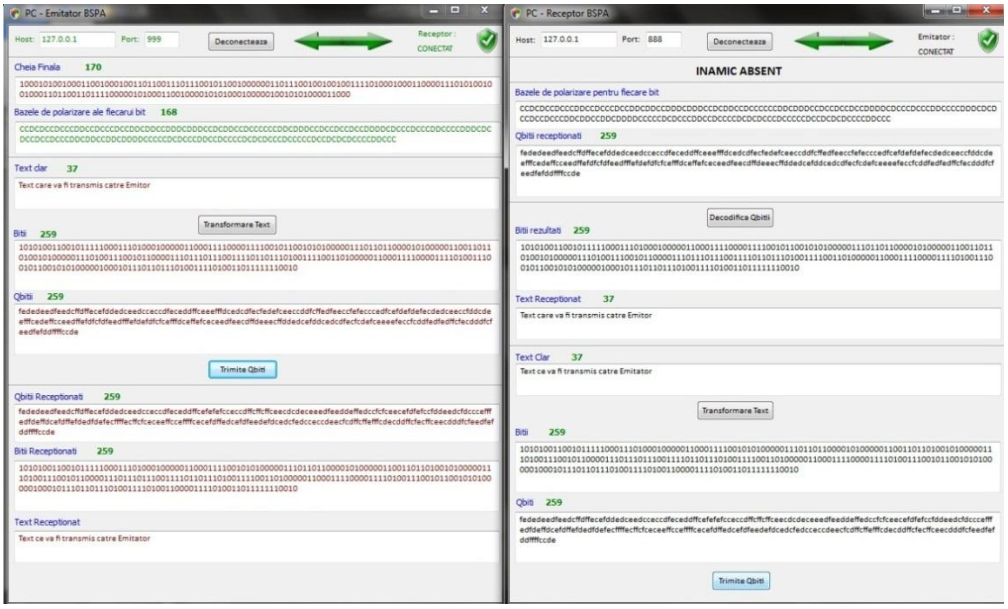

Figura 7.26. Detalii de transmisie în protocolul BSPA

7.6 Concluzii și contribuții

Programele de simulare realizate, simulează protocolul Base Selection and Polarization Agreement și un sistem de distribuire a cheilor cuantice realizat cu ajutorul algoritmului BB84, în diverse situații, cum ar fi în condiții ideale fără inamic, în condiții ideale cu inamic și în condiții ideale cu inamic dar folosind metoda QBTT de detectare a inamicului.

Comparând dimensiunea cheii finale, rezultată în urma rulării celor patru programe de simulare BB84 ideal, BB84 inamic, BB84 QBTT și BSPA, observăm că algoritmul BSPA ne oferă cea mai mare cheie iar metoda QBTT de detectare a inamicului îmbunătățește semnificativ dimensiunea cheii finale în cazul algoritmul BB84 cu inamic, tabelul 7.6 și figura 7.27.

Cheia inițială	Cheia finală BB84 ideal	Cheia finală BB84 inamic	Cheia finală BB84 QBTT	Cheia finală BSPA
320	159.6	116.8	157.8	318.7

Tabelul 7.6. Comparație între cheia inițială și cheia finală

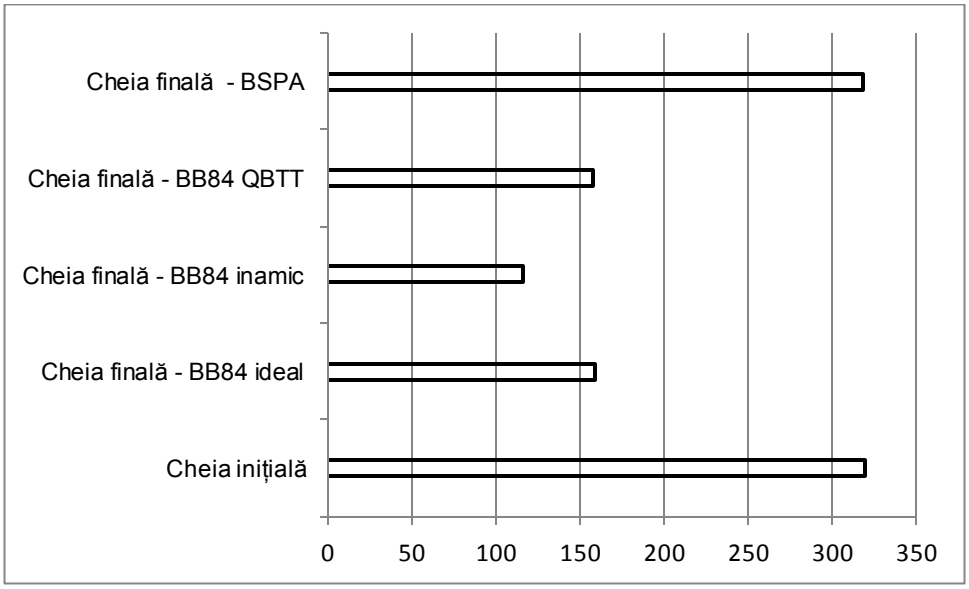

Figura 7.27. Comparație între cheia inițială și cheia finală

Se pot observa avantajele metodei QBTT de detectare a inamicului prin scăderea procentului de erori din cheia finală – QBER și avantajele protocolului cuantic BSPA prin reducerea aproape de 0 a procentului de erori din cheia finală, tabelul 7.7 și figura 7.28.

QBER % BB84 ideal	QBER % BB84 inamic	QBER % BB84 QBTT	QBER % BSPA
50.1	63.5	49.1	0.2

Tabelul 7.7. Comparație între QBER ale celor patru programe de simulare

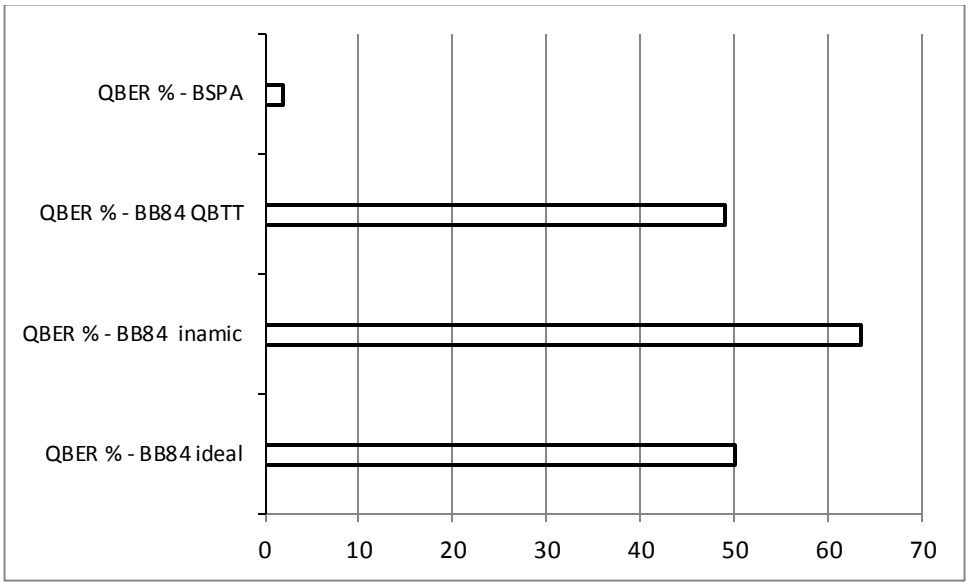

Figura 7.27. Comparație între QBER ale celor patru programe de simulare

Contribuțiile din acest capitol sunt :

- Proiectarea, realizarea și implementarea programelor de simulare a sistemelor de distribuire a cheilor cuantice.

Aceste contribuții sunt baza următoarelor lucrări științifice :

- Anghel C., „Research, Development and Simulation of Quantum Cryptographic Protocols", acceptată pentru publicare în revista „Electronics and Electrical Engineering".

8. Concluzii, contribuții și direcții de cercetare

8.1 Concluzii

Criptografia cuantică și în special sistemele de distribuire a cheilor cuantice – Quantum Key Distribution, realizează un schimb de chei cuantice între *emițător* și *receptor*. Acest schimb de chei cuantice se face în două etape, printr-un canal cuantic (fibră optică) și printr-un canal public (internet sau telefon). Cea mai importantă caracteristică a unui sistem de distribuire a cheilor cuantice și în special a criptografiei cuantice este că, orice încercare de interceptare a comunicației, nu numai că nu poate fi realizată, dar alertează și părțile legitime care comunică. Această proprietate a criptografiei cuantice, care nu poate fi obținută de criptografia clasică, îi conferă un standard foarte ridicat de securitate numit și *securitate necondiționată*.

Protocol Base Selection and Transmission Synchronization [3] utilizează *cheia finală* rezultată după etapele *Secret key reconciliation* și *Privacy amplification*, ale oricărui sistem de distribuire a cheilor cuantice. În funcție de *cheia finală*, *emițătorul* și *receptorul* vor stabili cu exactitate parametrii utilizați în transmisia cuantică, respectiv sistemul de baze de polarizare utilizat, tactul transmisiei și tipul de polarizare aplicat fiecărui foton în parte.

Datorită faptului că, încă din 1985, au fost descrise principiile de funcționare ale unui calculator cuantic [22], ne putem aștepta ca atunci când va fi operațional, algoritmul one-time pad să poată fi spart prin utilizarea forței brute. Acest lucru va face ca schemele de distribuire a cheilor cuantice să fie inutile, dar folosind algoritmul Base Selection and Transmission Synchronization – BSTS, care realizează o transmisie bidirecțională în totalitate cuantică vom putea comunica în secret.

Protocolul BSTS poate fi considerat un protocol de rețea cuantic între două calculatoare deoarece poate realiza o legătură bidirecțională între ele. Prin intermediul protocolului BSTS nu se dorește realizarea doar a unui schimb de chei cuantice, ca în cazul celorlalți algoritmi de distribuire a cheilor cuantice gen BB84, B92 sau E91, ci a se realiza o transmisie bidirecțională între cele două părți care comunică.

În concluzie putem spune că algoritmul Base Selection and Transmission Synchronization va realiza o transmisie respectiv recepție, fără erori datorită faptului că *emițătorul* și *receptorul* vor ști exact cum să polarizeze, respectiv să citească fiecare qbit în parte iar la sfârșitul transmisie se va realiza o corectare a erorilor.

Metoda Quantum Bit Travel Time – QBTT [5], prezintă avantajele că *inamicul* poate fi detectat „imediat" și cu „exactitate".

Receptorul poate detecta „imediat" prezența *inamicului* în timpul transmisiei cuantice, după fiecare qbit recepționat, prin măsurarea timpului parcurs de particula purtătoare de la *emițător* la *receptor*. În cazul altor algoritmilor de distribuire a cheilor cuantice, detectarea *inamicului* se face după finalizarea transmisiei cuantice, în etapa de comunicare a bazelor de polarizare pe canalul clasic, lăsând astfel în mâna *inamicului* informații importante legate de transmisie.

Inamicul poate fi depistat cu „exactitate" deoarece perturbările naturale apărute pe canalul cuantic nu produc întârzieri ale particulei purtătoare ci doar modificări ale polarizării. Această proprietate ne ajută să nu confundăm eventualele zgomotele de pe canalul cuantic cu prezența unui *inamic*. Alte metode de depistare a inamicului nu pot face diferența între zgomot și *inamic*.

Am văzut, în capitolul 3, că există anumite metode de atac [17,35,52], metode pur teoretice deocamdată, în care *inamicul*, chiar dacă interceptează transmisia, nu poate fi detectat prin metodele actuale și care pot compromite în totalitate transmisia cuantică dintre *emițător* și *receptor*. În schimb, prin utilizarea metodei Quantum Bit Travel Time, *inamicul* poate fi detectat imediat ce a intervenit pentru a citi un qbit, indiferent de metoda de atac folosită.

În concluzie putem spune că metoda Quantum Bit Travel Time - QBTT de detectare a inamicului poate fi implementată în oricare sistem de distribuire a cheilor cuantice și are avantajul că *inamicul* poate fi detectat cu *exactitate* și mai ales *imediat* după ce acesta a intervenit pentru a citi un qbit, în timpul transmisiei cuantice.

Prin implementarea metodei Quantum Bit Travel Time de detectare a inamicului în protocolul cuantic de comunicații Base Selection and Transmission Synchronization obținem un nou protocol cuantic de comunicații numit Base Selection and Polarization Agreement – BSPA [6], în care inamicul poate fi detectat în timpul transmisiei cuantice, după fiecare qbit recepționat și fără dubii că ar putea fi

confundat cu perturbări ale transmisiei deoarece zgomotele de pe canalul cuantic nu produc întârzieri ale particulei purtătoare.

Spre diferență de ceilalți algoritmi cuantici care au ca obiectiv doar schimbul cuantic de chei de criptare utilizate în combinație cu algoritmul de one-time pad, algoritmul BSPA realizează o transmisie bidirecțională în totalitate cuantică, rezultând un algoritm cuantic de comunicații între un emițător și un receptor.

Având în vedere faptul că un eventual *inamic* este practic depistat imediat ce încearcă să intercepteze transmisia cuantică, algoritmul BSPA realizează un protocol cuantic de comunicații care poate fi implementat pentru a realiza o rețea cuantică de comunicații.

În concluzie putem spune că implementând metoda de detectare a inamicului QBTT în protocolul BSTS obținem un nou protocol, protocolul BSPA, bidirecțional cuantic de comunicații care stabilește parametrii transmisiei și realizează corectarea erorilor.

Astfel, rulând programe de simulare, pentru o cheie inițială de 320 biți, a algoritmului BB84 în condiții ideale fără inamic, BB84 cu inamic, BB84 cu metoda QBTT de detectare a inamicului și cel al protocolului cuantic de comunicații BSPA, putem constata eficacitatea metodei QBTT de detectare a inamicului prin mărirea dimensiunii cheii finale respectiv scădera procentului de erori din cheia finală. Superioritatea protocolului cuantic BSPA se demonstrează prin obținerea unei chei finale de dimensiuni aproape identice cu cheia inițială și scăderea spre 0.2% a procentului de erori din cheia finală – QBER, tabelul 7.8 și figura 7.28.

Cheia inițială	Cheia finală BB84 ideal	Cheia finală BB84 inamic	Cheia finală BB84 QBTT	Cheia finală BSPA	QBER % BB84 ideal	QBER % BB84 inamic	QBER % BB84 QBTT	QBER % BSPA
320	162	110	148	320	49.4	65.6	52.2	0.0
320	161	114	150	317	49.7	64.4	51.6	0.9
320	152	129	167	320	52.5	59.7	46.3	0.0
320	172	131	157	319	46.3	59.1	49.4	0.3
320	158	126	159	320	50.6	60.6	48.8	0.0
320	153	105	152	317	52.2	67.2	50.9	0.9
320	165	105	168	320	48.4	67.2	45.9	0.0
320	163	108	157	319	49.1	66.3	49.4	0.3
320	150	117	158	320	53.1	63.4	49.1	0.0
320	160	123	162	319	50.0	61.6	47.8	0.3
320	159.6	116.8	157.8	318.7	50.1	63.5	49.1	0.2

Tabelul 8.1. Comparație între cele patru programe de simulare

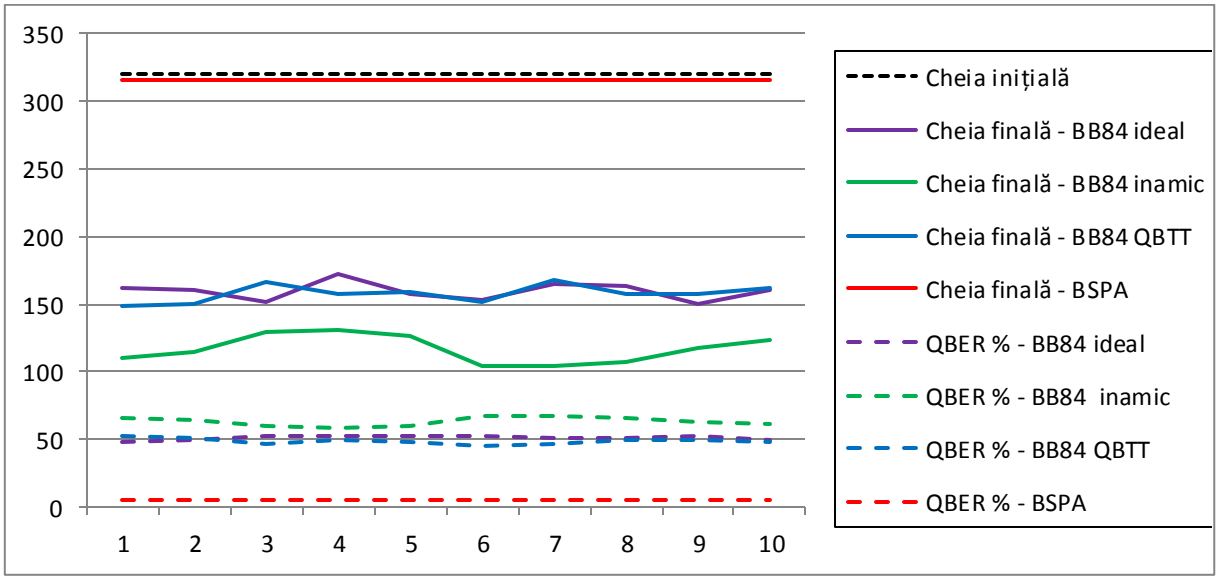

Figura 8.1. Comparație între cele patru programe de simulare

8.2 Contribuții

Algoritmii criptografici cuantici existenți realizează un schimb de chei, între un emițător și un receptor, rezultând la finalul transmisiei cuantice o cheie finală unică și secretă care va fi utilizată în coroborare cu algoritmul one-time pad [71,72], pentru a realiza o comunicare perfect sigură. În cazul acestor algoritmi transmisia cheii se realizează pe canalul cuantic iar comunicarea, cu algoritmul one-time pad, se realizează pe un canal clasic.

Una din contribuțiile aduse de această teză este proiectarea, realizarea și implementarea primului protocol de comunicații bidirecțional cuantic între două calculatoare, numit Base Selection and Transmission Synchronization, în care comunicația se realizează exclusiv pe un canal cuantic.

Prin etapa *base selection*, protocolul Base Selection and Transmission Synchronization stabilește cu exactitate ce baze de polarizare vor fi folosite și cum va fi polarizat respectiv citit fiecare qbit în parte.

Prin etapa *transmission synchronization*, protocolul Base Selection and Transmission Synchronization stabilește intervalul de timp în care *emițătorul* va transmite un foton real iar *receptorul* va citi fotonul respectiv, deoarece în restul intervalului *emițătorul* va transmite fotoni falși polarizați aleatoriu astfel încât *inamicul* să obțină mai mulți qbiți falși decât qbiți reali.

În figura 8.2 este prezentată diagrama funcțională a protocolului Base Selection and Transmission Synchronization.

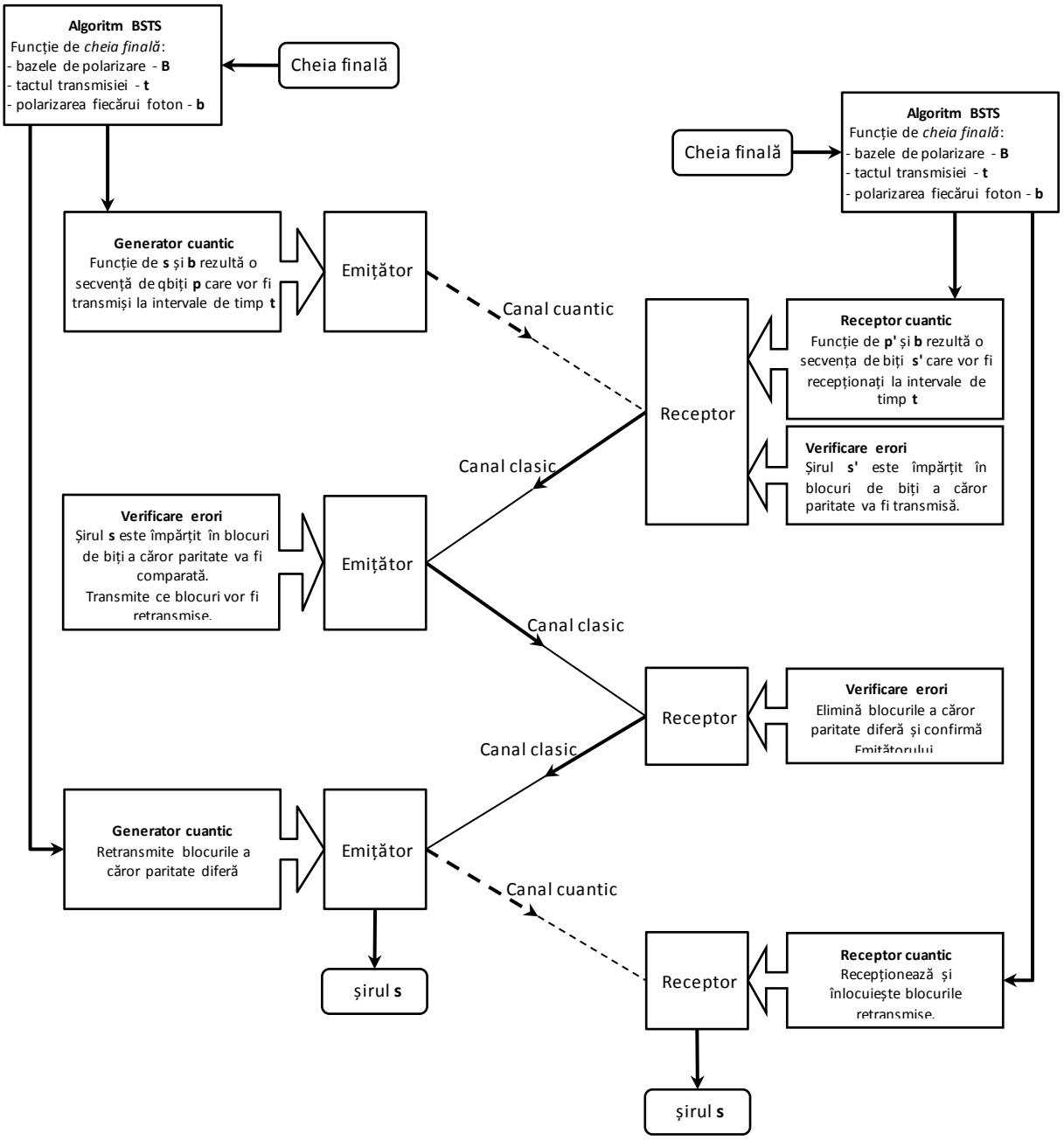

Figura 8.2. Diagrama funcțională a protocolului BSTS

Parametrii vizați de protocolul Base Selection and Transmission Synchronization sunt :

- ce pereche de baze de polarizare dintre cele trei, L (liniară), D (diagonală) și C (circulară), va fi folosită;
- intervalul de timp la care *emițătorul* va emite un qbit iar *receptorul* îl va citi;
- ce bază de polarizare va fi folosită pentru codificarea fiecărui bit în parte;

- verificarea biților recepționați la finalul transmisiei cuantice și retransmiterea blocurilor care conțin erori.

Metodele actuale de detectare a inamicului care interceptează un canal cuantic sunt ineficiente pentru unele metode de atac cum ar fi Photon Number Splitting, Man-In-the-Middle sau Quantum Cloning.

Altă contribuție adusă de această teză este proiectarea, realizarea și implementarea unei metode de detectare a interceptării care poate detecta prezența inamicului indiferent de metoda de atac folosită. Această metodă, numită metoda Quantum Bit Travel Time va realiza o detectare imediată și mai ales exactă a inamicului care încearcă să intercepteze canalul cuantic.

Detectarea exactă a *inamicului* se referă la faptul că acesta nu este confundat cu perturbările cauzate de canalul cuantic de comunicații.

Detectarea imediată a *inamicului* se referă la faptul că acesta este detectat în momentul în care a intervenit pentru a citi un qbit.

Metoda Quantum Bit Travel Time de detectare a interceptării poate fi implementată în orice sistem de distribuție a cheilor cuantice. Metoda se bazează pe faptul că filtrele de polarizare induc întârzieri ale particulei purtătoare [77]. Fiecare filtru aplicat va întârzia particula purtătoare cu o anumită perioadă de timp.

Dacă *inamicul* citește un qbit, adică aplică un filtru de polarizare, acesta va induce o întârziere suplimentară a particulei purtătoare, notată Δt, crescând astfel timpul parcurs de către aceasta de la *receptor* la *emițător*.

Inamicul poate fi detectat prin măsurarea timpului parcurs de particula purtătoare de la *emițător* la *receptor*, iar dacă $\Delta T' = \Delta T + \Delta t$ va rezulta că *inamicul* a fost prezent.

Inamicul este detectat indiferent de metoda de atac folosită deoarece orice metodă de atac presupune o întârziere a particulei purtătoare iar această întârziere va alerta *receptorul* de prezența *inamicului*.

În capitolul 7.4 este prezentat un program de simulare a unui sistem de distribuire a cheilor cuantice BB84 care folosește pentru detectarea inamicului metoda Quantum Bit Travel Time.

Comparând dimensiunea cheii finale și a procentului de erori din cheia finală – QBER dintre algoritmul BB84 cu inamic și algoritmul BB84 cu inamic dar utilizând pentru detectarea inamicului metoda Quantum Bit Travel Time, putem constata o

îmbunătățire semnificativă datorată în exclusivitate metodei QBTT, tabelul 8.2 și figura 8.3.

Cheia inițială	Cheia finală BB84 inamic	Cheia finală BB84 QBTT	QBER % BB84 inamic	QBER % BB84 QBTT
320	116.8	157.8	63.51	48.56

Tabelul 8.2. Avantajele metodei QBTT

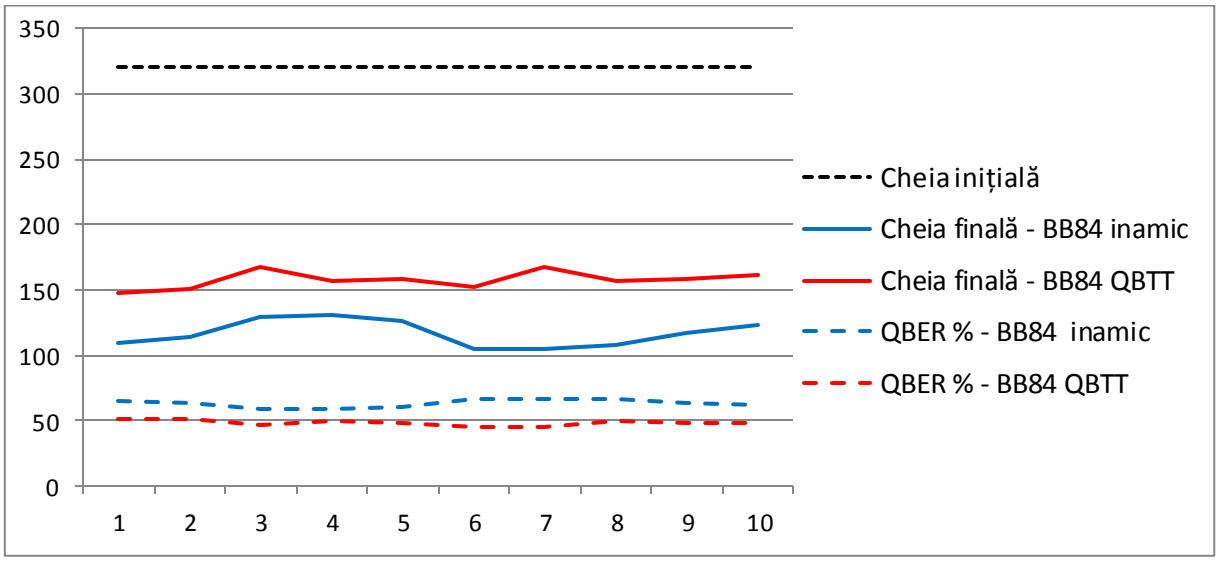

Figura 8.3. Avantajele motodei QBTT

O altă contribuție adusă de acestă teză este proiectarea, realizarea și implementarea unui protocol de comunicații bidirecțional cuantic între două calculatoare, numit Base Selection and Polarization Agreement, în care comunicația se realizează exclusiv pe un canal cuantic iar prezența inamicului este detectată imediat și cu exactitate indiferent de metoda de atac folosită.

Diagrama funcțională a protocolului Base Selection and Polarization Agreement este prezentată în figura 8.4.

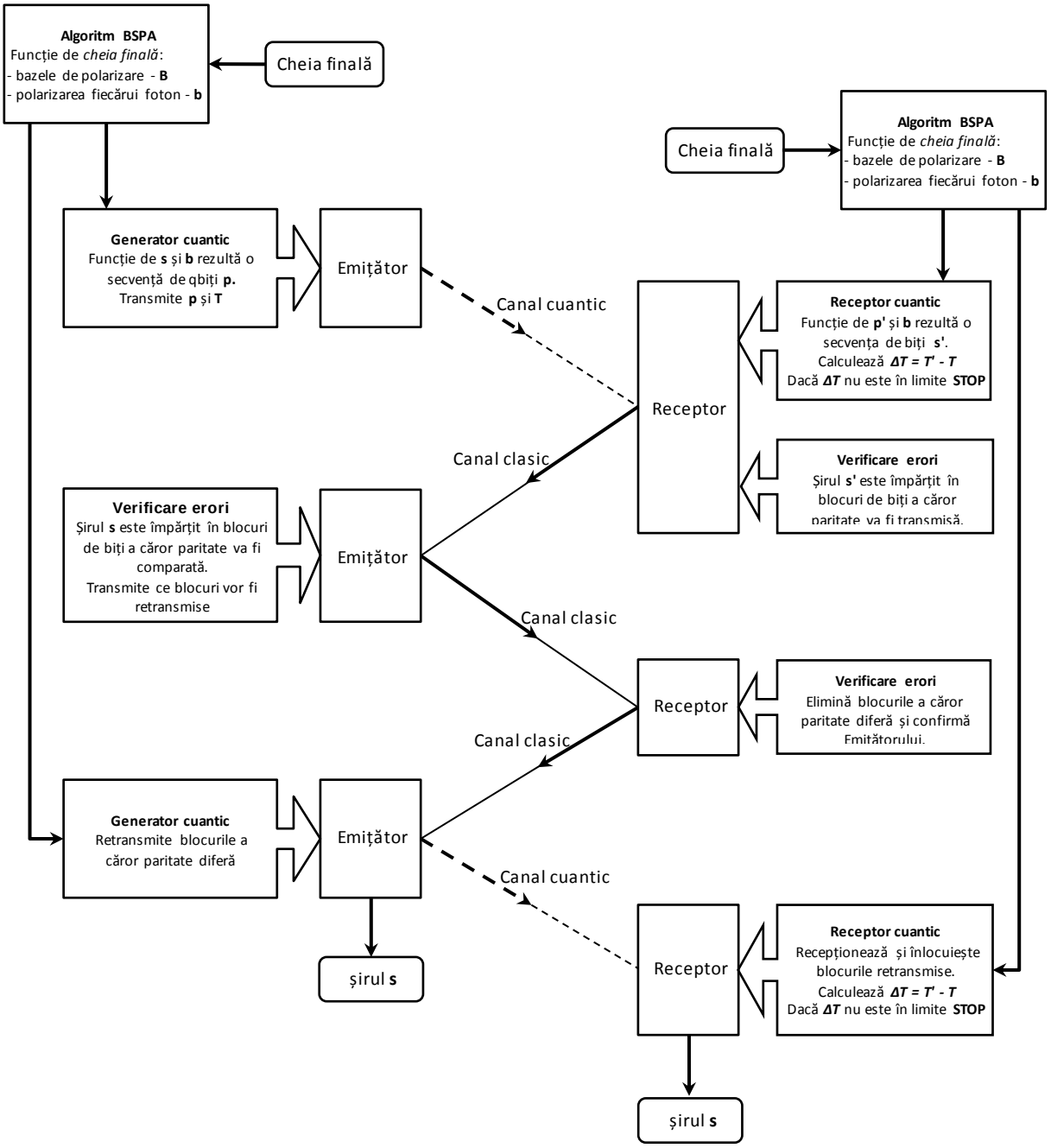

Figura 8.4. Diagrama functională a protocolului BSPA

Detectarea imediată a *inamicului* se referă la faptul că acesta este detectat în momentul în care a intervenit pentru a citi un qbit.

Detectarea exactă a *inamicului* se referă la faptul că acesta nu este confundat cu perturbările cauzate de canalul cuantic de comunicații.

Parametrii vizați de protocolul Base Selection and Polarization Agreement sunt :

- ce pereche de baze de polarizare dintre cele trei, L, D și C, va fi folosită;
- ce bază de polarizare va fi folosită pentru codificarea fiecărui bit în parte.

- detectarea *inamicului* prin monitorizarea timpului parcurs de către fiecare particulă purtătoare de la *emițător* la *receptor*.
- verificarea biților recepționați la finalul transmisiei cuantice și corectarea erorilor prin retransmiterea blocurilor care conțin inadvertențe.

În capitolul 7.5 este prezentat un programul de simulare a protocolului Base Selection and Polarization Agreement.

Comparând dimensiunea cheii finale și a procentului de erori din cheia finală – QBER dintre algoritmul BB84 cu metoda QBTT și protocolul Base Selection and Polarization Agreement, putem constata o îmbunătățire considerabilă a acestora în cazul algoritmului BSPA, tabelul 8.3 și figura 8.5.

Cheia inițială	Cheia finală BB84 QBTT	Cheia finală BSPA	QBER % BB84 QBTT	QBER % BSPA
320	157.8	319.1	48.6	0.2

Tabelul 8.3. Avantajele protocolului BSPA

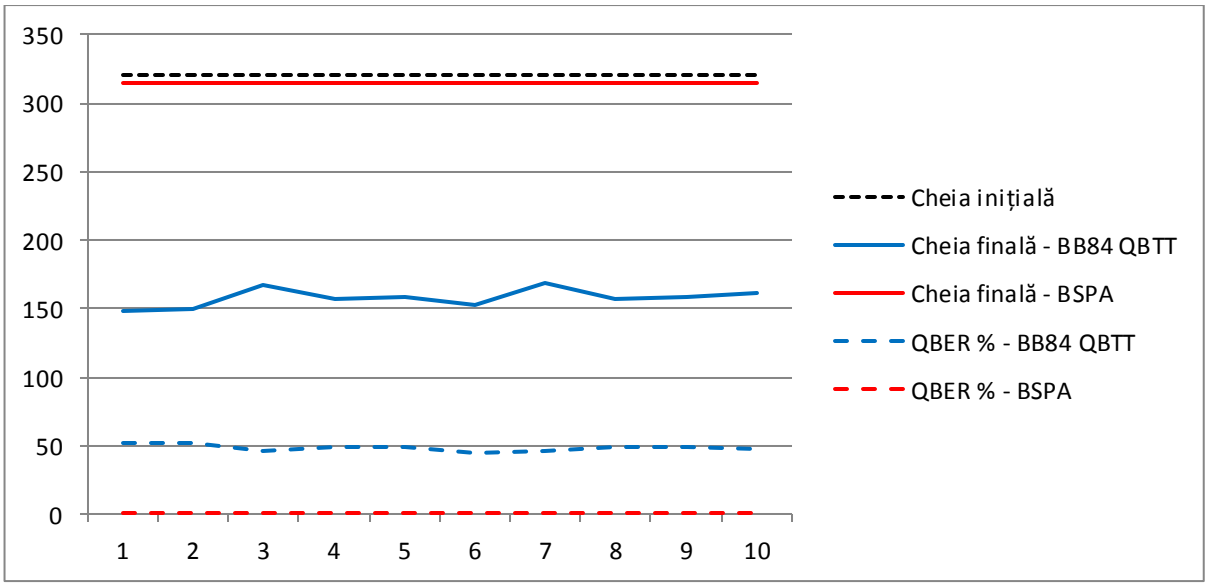

Figura 8.5. Avantajele protocolului BSPA

8.3 Diseminarea rezultatelor cercetării

Rezultatele cercetărilor doctorale au fost prezentate în 6 articole publicate sau acceptate spre publicare din care : 1 lucrare în revistă indexată ISI cu factor de impact 0.913, 2 lucrări în reviste indexate BDI, 2 lucrări la conferințe internaționale și 1 lucrare în biblioteca electronică Cornell University Library din SUA.

Reviste ISI cu factor de impact

- Anghel C., „Research, Development and Simulation of Quantum Cryptographic Protocols", acceptată pentru publicare în „Electronics and Electrical Engineering", (cotată ISI, factor de impact 0.913).

Reviste indexate BDI

- Anghel C., „New eavesdropper detection method in quantum cryptography", The annals of "Dunarea de Jos" University of Galati, fascicula III, vol.34, nr. 1, pag. 1-8, 2011.
- Anghel C., „New quantum cryptographic protocol", The annals of "Dunarea de Jos" University of Galati, fascicula III, vol.34, nr. 2, pag. 7-13, 2011.

Conferințe internaționale

- Anghel C., „Quantum cryptography algorithm", Proceedings ECIT2008 – 5th European Conference on Intelligent Systems and Technologies, Romania, Iasi, 2008.
- Anghel C. & Coman G., „Base selection and transmission synchronization algorithm in quantum cryptography", Proceedings CSCS17 - 17th International Conference on Control Systems and Computer Science, Romania, Bucharest, ISSN : 2066-4451, vol. 1, pag. 281-284, 2009.

Biblioteca electronică

- Anghel C., „Creșterea securității Sistemelor Informatice și de Comunicații prin Criptografia Cuantică", Cornell University Library, http://arxiv.org/, cite as: http://arxiv.org/abs/1006.5381v1, 2010.

8.4 Direcții de cercetare

Direcțiile de cercetare în domeniul criptografiei cuantice sunt concentrate pe mai multe planuri :

- mărirea dimensiunii cheii de criptare finale;
- securitatea algoritmilor de distribuire a cheilor cuantice;
- mărirea distanței dintre părțile care comunică;
- crearea unor algoritmi noi și îmbunătățirea celor existenți;
- utilizarea sateliților pentru realizarea transmisiei;
- realizarea unor repetoare cuantice;
- realizarea unei memorii cuantice;
- realizarea unor rețele cuantice;
- realizarea unui calculator cuantic.

8.4.1 Mărirea dimensiunii cheii finale

Principalul inconvenient al algoritmilor de distribuire a cheilor cuantice actuali este dimensiunea redusă a cheii finale comparativ cu dimensiunea inițială a cheii transmise. Astfel, accentul se pune pe realizarea unui algoritm de distribuire a cheilor cuantice care să permită obținerea de către cele două părți care comunică a unei chei unice, secrete de dimensiuni satisfăcătoare.

8.4.2 Securitatea QKD

Securitatea absolută a algoritmilor de distribuire a cheilor cuantice [12,30,37,43,48,55,61,66,68] este o problemă primordială în criptografia cuantică. Au fost propuse multe idei prin care un algoritm de distribuire a cheilor cuantice ar putea fi compromis [29,45,47,49-54,57,70,74,75,78], ceea ce impune găsirea unor algoritmi mai siguri, mai fiabili, mai stabili.

8.4.3 Mărirea distanței de transmisie

Distanța maximă până la care s-a putut realiza un schimb de chei cuantice este de aproximativ 250 km [65]. Pentru a fi eficient, un protocol de schimbare a

cheilor cuantice trebuie să nu fie limitat de distanța dintre părțile care comunică. Astfel, s-au născut mai multe idei prin care ar putea fi mărită această distanță prin utilizarea sateliților, utilizarea unor repetoare cuantice, utilizarea unor memorii cuantice sau realizarea unui calculator cuantic.

8.4.4 Crearea și îmbunătățirea algoritmilor

Una din preocupările permanente în domeniul criptografiei cuantice a fost, încă de la de descoperirea ei, crearea unor algoritmi noi, mai siguri și mai ușor de aplicat în practică. Totodată, o preocupare constantă este și îmbunătățirea algoritmilor existenți.

8.4.5 Utilizarea sateliților în QKD

Pentru mărirea distanței dintre părțile care comunică s-ar putea utiliza sateliții de comunicație [73]. Astfel, *Emițătorul* poate transmite către satelit secvența de qbiți iar în momentul în care satelitul ajunge în zona *Receptorului* să-i retransmită acestuia qbiții.

8.4.6 Repetoare cuantice

Pierderile de semnal de pe fibra optică, datorate distanțelor mari dintre cele două părți care comunică, limitează schimbul de chei cuantice. Pentru a realiza schimbul de chei cuantice la distanțe foarte mari, fără a avea încredere în receptori intermediari, avem nevoie de repetoare cuantice [18,60]. Aceste repetoare cuantice vor realiza schimbul de qbiți fără ai măsura și astfel fără ai modifica.

8.4.7 Memorii cuantice

Memoriile cuantice [64] sunt elemente foarte importante pentru sistemele informatice cuantice cum ar fi rețelele cuantice [11,41], repetoarele cuantice [18,60] și pentru calculatoarele cuantice [42,44]

Aceste memorii cuantice trebuie să realizeze doar stocarea qbiților fără ai măsura deoarece orice încercare de măsurare a unui qbit înseamnă automat și modificarea lui [33], datorită principiului incertitudinii al lui Heisenberg.

8.4.8 Rețele QKD

Un alt aspect de care trebuie ținut cont este realizarea unor rețele cuantice [11,41]. Aceste rețele ar avea ca suport fizic fibra optică iar purtătorii informației ar fi fotonii polarizați – qbiții.

8.4.9 Calculatoare cuantice

Ultimul pas ar fi realizarea unui calculator cuantic. Având la dispoziție repetoare cuantice [18,60] și memorii cuantice [64], pentru realizarea unui calculator cuantic ar mai fi necesar un microprocesor cuantic [23]. Realizarea unui calculator cuantic ar rezolva toate problemele existente, legate de viteza de procesare și mai ales de securizarea informațiilor.

BIBLIOGRAFIE

[1]. Alleaume R., et al., „SECOQC White Paper on Quantum Key Distribution and Cryptography", arXiv:quant-ph/0701168v1, 2007.

[2]. Anghel C., „Quantum cryptography algorithm", Proceedings ECIT2008 – 5th European Conference on Intelligent Systems and Technologies, Romania, Iasi, 2008.

[3]. Anghel C. & Coman G., „Base selection and transmission synchronization algorithm in quantum cryptography", Proceedings CSCS17 - 17th International Conference on Control Systems and Computer Science, Romania, Bucharest, ISSN : 2066-4451, vol. 1, pag. 281-284, 2009.

[4]. Anghel C., „Creșterea securității Sistemelor Informatice și de Comunicații prin Criptografia Cuantică", Cornell University Library, http://arxiv.org/, cite as: http://arxiv.org/abs/1006.5381v1, 2010.

[5]. Anghel C., „New eavesdropper detection method in quantum cryptography", The annals of "Dunarea de Jos" University of Galati, fascicula III, vol.34, nr. 1, pag. 1-8, 2011.

[6]. Anghel C., „New quantum cryptographic protocol", The annals of "Dunarea de Jos" University of Galati, fascicula III, vol.34, nr. 2, pag. 7-13, 2011.

[7]. Anghel C., „Research, Development and Simulation of Quantum Cryptographic Protocols", acceptată pentru publicare în revista „Electronics and Electrical Engineering".

[8]. Bell J.S., „On the Einstein-Podolsky-Rosen paradox", Physics 1, vol. 1, nr. 3, pag. 195-200, 1964.

[9]. Bennett C.H., Brassards G. & Jean-Marc R., „Privacy amplification by public discussions", Siam Journal on Computing, vol. 17, nr. 2, pag. 210-229, 1988.

[10]. Bennett C.H. & Brassard G., „Quantum cryptography : Public key distribution and coin tossing", Proceedings of IEEE International Conference on Computers Systems and Signal Processing, Bangalore India, pag. 175-179, 1984.

[11]. Bennett C.H., Bessette F., Brassard G., Salvail L., & Smolin J., „Experimental quantum cryptography", Journal of Cryptology, vol. 5, no. 1, pag. 3-28, 1992.

[12]. Bennett C.H., „Quantum cryptography using any two nonorthogonal states", Physical Review Letters, vol. 68, pag. 3121-3124, 1992.

[13]. Bouwmeester D., Ekert A. & Zeilinger A., „The Physics of Quantum Information. Quantum Cryptography, Quantum Teleportation, Quantum Computation", Springer, pag. 314, 2000.

[14]. Branciard C., Gisin N., Kraus B. & Scarani V., „Security of two quantum cryptography protocols using the same four qubit states", Physical Review A, vol. 72, nr. 3, 2005.

[15]. Brassard G. & Salvail L., „Secret-key reconciliation by public discussion", Proceeding - EUROCRYPT '93, Lecture Notes in Computer Science, vol. 765, pag. 410-423, 1994.

[16]. Brassard G., Lutkenhaus N., Mor T. & Sanders B., „Limitations on practical quantum cryptography", Physical Review Letters, vol. 85, nr. 6, pag. 1330-1333, 2000.

[17]. Brassard G., Lütkenhaus N., Mor T & Sanders B.C., „Limitation on practical quantum cryptography", Physical Review Letters, Vol. 85, pag. 1330-1333, 2000.

[18]. Briegel H.J., Dür W., Cirac J.I. & Zoller P., „ Quantum Repeaters: The Role of Imperfect Local Operations in Quantum Communication", Physical Review Letters, vol. 81, 1998.

[19]. Bruss D., „Optimal Eavesdropping in Quantum Cryptography with Six States", Physical Review Letters, vol. 81, nr. 14, pag. 3018-3021, 1998.

[20]. De Burgh M. & Bartlett S.D., „Quantum methods for clock synchronization: Beating the standard quantum limit without entanglement", Physical Review A, vol. 72, 2005.

[21]. Desurvire E., „Classical and Quantum Information Theory", Cambridge University Press, vol. 691, 2009.

[22]. Deutsch D., „Quantum theory, the Church-Turing principle and the universal quantum computer", Proceedings of the Royal Society of London A, vol. 400, pag. 97-117, 1985.

[23]. Dewes A., Lauro R., et al., „Demonstrating quantum speed-up in a superconducting two-qubit processor", arXiv:1110.5170v1, 2011.

[24]. Durt T., Kaszlikowski D., Chen J.L. & Kwek L.C., „Security of quantum key distributions with entangled qudits", Physical Review A, vol. 69, nr. 3, pag. 1-11, 2004.

[25]. Einstein A., Podolsky B. & Rosen N., „Can quantum, mechanical description of physical reality be considered complete?", Physical Review, vol. 47, pag. 777-780, 1935.

[26]. Ekert A., „Quantum cryptography based on Bell's theorem", Physical Review Letters, vol. 67, nr. 6, pag. 661-663, 1991.

[27]. Gisin N. & Huttner B., „Quantum cloning, eavesdropping and Bell's inequality", Physics Letters A, vol. 228, pag. 13-21, 1997.

[28]. Gisin N., Ribordy G., Tittle W. & Zbinden H., "Quantum cryptography", Reviews of Modern Physics, vol. 74, 2002.

[29]. Gisin N., Fasel S., Krauss B., Zbinden H. & Ribordy G., „Trojan-horse attacks on quantum-key-distribution systems", Physical Review A, vol. 73, 2006.

[30]. Gottesman D., Lo H.K., Lütkenhaus N. & Preskill J., „Security of Quantum Key Distribution with Imperfect Devices", Quantum Information & Computation, vol. 4, pag. 325, 2004.

[31]. Gupta N.L., Mehrotra D.R. & Saxena A., „Quantum cryptographic protocols for secure communication", Journal of Computer Science, vol. 8, nr.1, pag. 65-74, 2009.

[32]. Harrelson C. & Kerenidis I., „Quantum clock synchronization with one qubit", http://arxiv.org/abs/cs/0103021v3, 2001.

[33]. Heisenberg W., „About the perceptual content of quantum kinematics and mechanics", Journal of Physics, vol. 43, pag. 172-198, 1927.

[34]. Huang D., Chen Z., Guo Y. & Lee M., „Quantum secure direct communication based on chaos with authentication", Journal of the Physical Society of Japan, vol. 76, 2007.

[35]. Huttner B., Imoto N., Gisin N. & Mor T., „Quantum cryptography with coherent states", Physical Review A, vol. 51, pag. 1863-1869, 1995.

[36]. Inamori H., Rallan L. & Vedral V., „Security of EPR-based quantum cryptography against incoherent symmetric attacks", Journal of Physics A, vol. 34, nr. 35, pag. 6913-6918, 2001.

[37]. Inamori H., Lütkenhaus N. & Mayers D., „Unconditional Security of Practical Quantum Key Distribution", European Physical Journal D, vol. 41, pag. 599-627, 2007.

[38]. Jozsa R., Abrams D.S., Dowling J.P. & Williams C.P., „Quantum Clock Synchronization Based on Shared Prior Entanglement", Physical Review Letters, vol. 85, pag. 2010-2013, 2000.

[39]. Kahn D., „The Codebreakers - The Story of Secret Writing", (ISBN 0-684-83130-9), 1974.

[40]. Kaszlikowski D., Christandl M. et al., „Quantum cryptography based on qutrit Bell inequalities", Physical Review A, vol. 67, nr. 1, 2003.

[41]. Kimble H.J., „The quantum internet", Nature, vol. 453, pag. 1023-1030, 2008.

[42]. Knill E., Laflamme R. & Milburn G.J., „A scheme for efficient quantum computation with linear optics", Nature, vol. 409, pag. 46-52, 2001.

[43]. Koashi M. & Preskill J., „Secure quantum key distribution with an uncharacterized source", Physical Review Letters, vol. 90, 2003.

[44]. Kok P., Munro W.J., Nemoto K., Ralph T.C., Dowling J.P. & Milburn G.J., „Linear optical quantum computing", Reviews of Modern Physics, vol. 79, pag. 135-174, 2007.

[45]. Kurtsiefer C., Zarda P., Mayer S. & Weinfurter H., „The breakdown flash of silicon avalanche photodiodes – backdoor for eavesdropper attacks", Journal of Modern Optics, vol. 48, pag. 2039-2047, 2001.

[46]. Kwiat P.G., Mattle K., Weinfurter H., Zeilinger A., Sergienko A.V. & Shih Y., „New High-Intensity Source of Polarization-Entangled Photon Pairs", Physical Review Letters, nr. 75, pag. 4337, 1995.

[47]. Lamas-Linares A. & Kurtsiefer C., „Breaking a quantum key distribution system through a timing side channel", Optics Express, vol. 15, pag. 9388-9393, 2007.

[48]. Lomonaco S.J., „A Quick Glance at Quantum Cryptography", Journal Cryptologia, vol. 23, pag. 1-41, 1999.

[49]. Lydersen L. et al., „Hacking commercial quantum cryptography systems by tailored bright illumination", Nature Photonics, vol. 4, pag. 686-689, 2010.

[50]. Lydersen L. et al., „Thermal blinding of gated detectors in quantum cryptography", Optics Express, vol. 18, pag. 27938-27954, 2010.

[51]. Makarov V. & Hjelme D.R., „Faked states attack on quantum cryptosystems", Journal of Modern Optics, vol. 52, pag. 691-705, 2005.

[52]. Makarov V., Anisimov A. & Skaar J., „Effects of detector efficiency mismatch on security of quantum cryptosystems", Physical Review A, vol. 74, pag. 1-11, 2005.

[53]. Makarov V., „Controlling passively quenched single photon detectors by bright light", New Journal of Physics, vol. 11, 2009.

[54]. Makarov V., Anisimov A. & Sauge S., „Quantum hacking: adding a commercial actively-quenched module to the list of single-photon detectors controllable by Eve", Nature Photonics, vol. 4, pag. 686-689, 2010.

[55]. Mayers D., „Advances in cryptology", Proceedings of Crypto´96, vol. 1109, pag. 343-357, 1996.

[56]. Muhammad N.A. & Zuriati A.Z., „Implemetation of BB84 Quantum Key Distribution Protocol's with Attacks", European Journal of Scientific Research, vol. 32, nr.4, pag. 460-466, 2009.

[57]. Nauerth S., Fürst M., Schmitt-Manderbach T.,Weier H. & Weinfurter H., „Information leakage via side channels in free space BB84 quantum cryptography", New Journal of Physics, vol. 11, 2009.

[58]. Nielsen M.A. & Chuang I.L., „Quantum Computation and Quantum Information", Cambridge University Press, 2000.

[59]. Rivest R.L., Shamir A. & Adleman L., „A method for obtaining digital signatures and public-key cryptosystems", Communications of the ACM, vol. 21, pag. 120-126, 1978.

[60]. Sangouard N., Simon C., De Riedmatten H. & Gisin N., „Quantum repeaters based on atomic ensembles and linear optics", Reviews of Modern Physics, vol. 83, pag. 33-80, 2011.

[61]. Scarani V., Bechmann-Pasquinucci H., Cerf N.J. et al., „The security of practical quantum key distribution", Review of Modern Physics, vol. 81, pag. 1301-1350, 2009.

[62]. Shannon C., „Communication theory of secrecy systems", Bell System Technical Journal, vol. 28, pag. 656-715, 1949.

[63]. Simon J.P., & Townsend P.D., „Quantum cryptography: how to beat the code breakers using quantum mechanics", Comtemporay Physics, vol. 36, nr. 3, pag. 165-195, 1995.

[64]. Simon C., Afzelius M., et al., „Quantum Memories. A Review based on the European Integrated Project 'Qubit Applications (QAP)'", European Physical Journal D, vol. 58, nr. 1, pag. 1-22, 2010.

[65]. Stucki D. et al., „High rate, long-distance quantum key distribution over 250 km of ultra low loss fibres", New Journal of Physics, vol. 11, 2009.

[66]. Townsend P.D., Rarity J.G. & Tapster P.R., „Single photon interference in 10km long optical fibre interferometer", Electronic Letters, vol. 29, pag. 634-635, 1993.

[67]. Townsend P.D., „Secure key distribution system based on quantum cryptography", Electronic Letters, vol. 30, nr. 10, pag. 809-811, 1994.

[68]. Townsend P.D. & Thompson I., „A quantum key distribution channel based on optical fibre", Journal of Modern Optics, vol. 41, nr. 12, pag. 2425-2433, 1994.

[69]. Treiber A., „A fully automated quantum cryptography system based on entanglement for optical fibre networks", New Journal of Physics, vol. 11, 2009.

[70]. Vakhitov A., Makarov V. & Hjelme D.R., „Large pulse attack as a method of conventional optical eaves-dropping in quantum cryptography", Journal of Modern Optics, vol. 48, pag. 2023-2038, 2001.

[71]. Vernam G., „Secret signaling system", U.S. patent No. 1310719, 1919.

[72]. Vernam G., „Cipher printing telegraph system for secret wire and radio telegraphic communications," Journal of IEEE, vol. 55, pag. 109-115, 1926.

[73]. Villoresi P., et al., „Space-to-ground quantum-communication using an optical ground station : a feasibility study", Proceedings of SPIE, vol. 5551, pag. 113-120, 2004.

[74]. Wiechers C. et al., „After-gate attack on a quantum cryptosystem", New Journal of Physics, vol. 13, 2010.

[75]. Xu F., Qi B. & Lo H.K., „Experimental demonstration of phase-remapping attack in a practical quantum key distribution system", New Journal of Physics, vol. 12, 2010.

[76]. Zeilinger A., „Dance of the Photons: From Einstein to Quantum Teleportation", pag. 205, New York, 2010.

[77]. Zhao S. & De Raedt H., „Event-by-event Simulation of Quantum Cryptography Protocols", Journal of Computational and Theoretical Nanoscience, vol. 5, nr. 4, pag. 490-504, 2008.

[78]. Zhao Y., Fung C.H.F., Qi B., Cheng C. & Lo, H.-K., „Quantum hacking: Experimental demonstration of time-shift attack against practical quantum-key- distribution systems", Physical Review A, vol. 78, 2008.

[79]. Zhang Z., Liu J., Wang D. & Shi S., „Quantum direct communication with authentication", Physical Review A, vol. 75, 2007.

Anexa 1 - Codul sursă al programelor

Programul BB84 cu metoda QBTT

Emițător.cpp

```
//---------------------------------------------------------------------------
#include <vcl.h>
#include <time.h>
#include <MMSYSTEM.H>
#include <sys/timeb.h>
#include <stdio.h>
#pragma hdrstop
#include "Emitator.h"
#include "DetaliiE.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma link "IdCustomTCPServer"
#pragma link "IdTCPServer"
#pragma link "IdContext"
#pragma link "IdSimpleServer"
#pragma link "trayicon"
#pragma resource "*.dfm"
TFormEmitator *FormEmitator;
//---------------------------------------------------------------------------
__fastcall TFormEmitator::TFormEmitator(TComponent* Owner) : TForm(Owner) {
}
//---------------------------------------------------------------------------
char baseE[320] = "";
char cheieE[320] = "";
char qbitsE[320] = "";
char finalkeyE;
char baseI[320] = "";
char IqbitsR[320];
UnicodeString EFinalKey = "";
UnicodeString CheiaR = "";
void __fastcall TFormEmitator::FormShow(TObject *Sender)
{
  NrBitiCheiaBrutaE->Text = "0";
  NrBitiEmitator->Text = "0";
  NrBitiCheiaFinalaE->Text = "0";
  ProcentCheiaBruta->Text = "0";
  ProcentCheiaFinala->Text = "0";
  char c[1]="";
  randomize();
```

```
 int nr;
 for (int i = 0; i < 320; i++) {
        switch (random(2)) {
               case 0 :
                      baseE[i] = 'L';
                      break;
               case 1 :
                      baseE[i] = 'D';
                      break;
        }
 }
 EBasesMemo->Text = baseE;
 for (int i = 0; i < 320; i++) {
        itoa(random(2), c, 10);
        cheieE[i] = *c;
 }
 ESecretKeyMemo->Text = AnsiString(cheieE, 320);
 NrBitiEmitator->Text = ESecretKeyMemo->GetTextLen();
}
//---------------------------------------------------------------------------
void __fastcall TFormEmitator::ButtonTrimiteQbitsClick(TObject *Sender)
{
 Word Hour1, Min1, Sec1, MSec1;
 UnicodeString TStamp, head = "Q";
 if(!IdTCPClientE_1->Connected()) {
          ShowMessage("Conexiune nerealizata !!");
          return;
 }
for (int i = 0; i < 320; i++) {
        if ((baseE[i] == 'L') && (cheieE[i] == '0'))
               qbitsE[i] = 'a';
        else if ((baseE[i] == 'L') && (cheieE[i] == '1'))
               qbitsE[i] = 'b';
        if ((baseE[i] == 'D') && (cheieE[i] == '0'))
               qbitsE[i] = 'c';
        else if ((baseE[i] == 'D') && (cheieE[i] == '1'))
               qbitsE[i] = 'd';
        TDateTime time1 = Now();
        DecodeTime(time1, Hour1, Min1, Sec1, MSec1);
        TStamp = IntToStr(Hour1)+ "." + IntToStr(Min1)+ "." + IntToStr(Sec1)
          + "." + IntToStr(MSec1) + ".";
        Memo1->Text = TStamp;
        Sleep(100);
        IdTCPClientE_1->Socket->WriteLn(head + TStamp + qbitsE[i]);
 }
 EQbitsMemo->Text = qbitsE;
}
//---------------------------------------------------------------------------
```

```
void __fastcall TFormEmitator::IdTCPServerE_2Execute(TIdContext  *AContext)
{
 UnicodeString head = "T";
 UnicodeString c = AContext->Connection->IOHandler->ReadLn();
 UnicodeString header = c.SubString(1,1);
 if (header == "B") {
        int i = StrToInt(c.SubString(3,c.Length()-1));
        UnicodeString ch = c.SubString(2,1);
        UnicodeString cc = baseE[i];
        if( (i >= 0) && (i < 320) && (cc == ch) ) {
                IdTCPClientE_1->IOHandler->WriteLn(head + "1" + IntToStr(i));
                EFinalBases->Text = EFinalBases->Text + ch;
                EFinalQbits->Text = EFinalQbits->Text + qbitsE[i];
                switch (qbitsE[i])  {
                 case 'a' :
                                finalkeyE = '0';
                                break;
                 case 'b' :
                                finalkeyE = '1';
                                break;
                 case 'c' :
                                finalkeyE = '0';
                                break;
                 case 'd' :
                                finalkeyE = '1';
                                break;
                }
                EFinalKey += finalkeyE;
                EFK->Text = EFK->Text + finalkeyE;
                AContext->Connection->IOHandler->Write(Byte(1));
        }
        else {
                IdTCPClientE_1->IOHandler->WriteLn(head + "0" + IntToStr(i));
                AContext->Connection->IOHandler->Write(Byte(0));
                EFinalBases->Text = EFinalBases->Text + '*';
                EFinalKey += 'x';
        }
        NrBitiCheiaBrutaE->Text = EFK->GetTextLen();
        ProcentCheiaBruta->Text=EFK->GetTextLen()*100/ESecretKeyMemo  ->GetTextLen();
 }
 else  {
        CheiaR = c.SubString(2,c.Length());
        for (int i = 0; i <= CheiaR.Length(); i++)        {
          if (CheiaR.SubString(i+1,1) == EFK->Text.SubString(i+1,1))   {
                   CheiaFinala->Text = CheiaFinala->Text + EFK->Text.SubString(i+1,1);
          }
        }
        NrBitiCheiaFinalaE->Text = CheiaFinala->GetTextLen();
```

```
        ProcentCheiaFinala->Text=CheiaFinala->GetTextLen()*100/ESecretKeyMemo-
>GetTextLen();
        AContext->Connection->IOHandler->WriteLn(EFK->Text);
  }
}
//---------------------------------------------------------------------------
void __fastcall TFormEmitator::ButonTrimiteTextCriptatClick(TObject  *Sender)
{
  if(!IdTCPClientE_1->Connected()) {
          ShowMessage("Conexiune nerealizata !!");
          return;
   }
   UnicodeString TextCriptat = "";
   UnicodeString CF = "", TC = "", cc;
   char ch1, ch2;
   int nr1, nr2, nr3;
   CF = CheiaFinala->Text;
   TC = TextClar->Text;
   int j = 1;
   for (int i = 0; i <= TC.Length(); i++) {
          cc = TC.SubString(i+1,1);
          ch1 = *cc.t_str();
          nr1 = (int) ch1;
          nr2 = StrToInt(CF.SubString(j,3));
          nr3 = nr2 + nr1;
          if (nr3 < 10) {
                TextCriptat = TextCriptat + "00" + nr3;
          } else if (nr3 < 100) {
                  TextCriptat = TextCriptat + "0" + nr3;
                } else {
                        TextCriptat = TextCriptat + nr3;
                  }
          j += 3;
   }
   UnicodeString headC = "C";
   IdTCPClientE_1->Socket->WriteLn(headC + TextCriptat);
}
//---------------------------------------------------------------------------
void __fastcall TFormEmitator::ButtonConectareClick(TObject  *Sender)
{
        try {
                if(ButtonConectare->Caption == "Conecteaza") {
                        IdTCPClientE_1->Host = EditHost->Text;
                        IdTCPClientE_1->Port = StrToInt(EditPort->Text);
                        IdTCPClientE_1->Connect();
                        if(IdTCPClientE_1->Connected()) {
                                ButtonConectare->Caption = "Deconecteaza";
                                Image2->Visible = True;
```

```
                        Image4->Visible = False;
                    }
                    return;
                }
                if(ButtonConectare->Caption == "Deconecteaza") {
                    if(IdTCPClientE_1->Connected())
                        IdTCPClientE_1->Disconnect();
                    if(!IdTCPClientE_1->Connected()) {
                            ButtonConectare->Caption = "Conecteaza";
                            Image2->Visible = False;
                            Image4->Visible = True;
                    }
                    return;
                }
        } catch (EIdException &e) {
                ShowMessage(e.Message);
          }
}
//---------------------------------------------------------------------------
void __fastcall TFormEmitator::IdTCPServerE_2Disconnect(TIdContext  *AContext)
{
        if(Label6->Caption == "CONECTAT")
                if(!(AContext->Connection->IOHandler->Connected())) {
                    Label6->Caption = "NECONECTAT";
                    ImgC->Visible = False;
                    ImgN->Visible = True;
                    Image1->Visible = False;
                    Image3->Visible = True;
                }
}
//---------------------------------------------------------------------------
void __fastcall TFormEmitator::ButtonDetaliiClick(TObject *Sender)
{
  FormDetaliiE = new TFormDetaliiE(Application);
  FormDetaliiE->Show();
  FormDetaliiE->EFinalKeyDetaliiR->Text = "";
  FormDetaliiE->EFinalKeyDetaliiR->Text = EFinalKey;
  FormDetaliiE->RFinalKeyDetaliiE->Text = CheiaR;
  FormDetaliiE->EFinalBases->Text = EFinalBases->Text;
}
//---------------------------------------------------------------------------
void __fastcall TFormEmitator::IdTCPServerE_2Connect(TIdContext  *AContext)
{
        if(Label6->Caption == "NECONECTAT")
                if(AContext->Connection->IOHandler->Connected()) {
                    Label6->Caption = "CONECTAT";
                    ImgC->Visible = True;
                    ImgN->Visible = False;
```

```
                        Image1->Visible = True;
                        Image3->Visible = False;
                }
}
//---------------------------------------------------------------------------
```

Receptor.cpp

```
//---------------------------------------------------------------------------
#include <vcl.h>
#pragma hdrstop
#include "Receptor.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma link "IdTCPClient"
#pragma link "IdTCPConnection"
#pragma link "IdIcmpClient"
#pragma link "IdRawBase"
#pragma link "IdRawClient"
#pragma resource "*.dfm"
TFormReceptor  *FormReceptor;
//---------------------------------------------------------------------------
__fastcall TFormReceptor::TFormReceptor(TComponent*  Owner)  : TForm(Owner)
{
}
//---------------------------------------------------------------------------
char baseR[320] = {0};
char finalkeyR;
char RqbitsI[320];
int count = 0;
void __fastcall TFormReceptor::FormShow(TObject  *Sender)
{
  NrBitiCheiaFinala->Text = "0";
  NrQbitiR->Text = "0";
  NrBitiCheiaBruta->Text = "0";
  randomize();
  for (int i = 0; i < 320; i++) {
        switch (random(2)) {
                case 0 :
                        baseR[i] = 'L';
                        break;
                case 1 :
                        baseR[i] = 'D';
                        break;
        }
        RBasesMemo->Text = AnsiString(baseR, 320);
```

```
  }
}
//---------------------------------------------------------------------------
void __fastcall TFormReceptor::ButtonConectareRClick(TObject *Sender)
{
 Try {
                if(ButtonConectareR->Caption == "Conecteaza") {
                        IdTCPClientR_2->Host = EditHostR->Text;
                        IdTCPClientR_2->Port = StrToInt(EditPortR->Text);
                        IdTCPClientR_2->Connect();
                        if(IdTCPClientR_2->Connected()) {
                                ButtonConectareR->Caption = "Deconecteaza";
                                Image1->Visible = True;
                                Image3->Visible = False;
                        }
                        return;
                }
                if(ButtonConectareR->Caption == "Deconecteaza") {
                        if(IdTCPClientR_2->Connected())
                                IdTCPClientR_2->Disconnect();
                        if(!IdTCPClientR_2->Connected()) {
                                ButtonConectareR->Caption = "Conecteaza";
                                Image1->Visible = False;
                                Image3->Visible = True;
                        }
                        return;
                }
        }
        catch (EIdException &e)       {
                ShowMessage(e.Message);
        }
}
//---------------------------------------------------------------------------
void __fastcall TFormReceptor::TrimiteBazeClick(TObject *Sender)
{
  UnicodeString str;
  UnicodeString ch, header;
  header = "B";
  if(!IdTCPClientR_2->Connected())   {
        ShowMessage("Conexiune nerealizata !!");
        return;
  }

  RFinalBases->Text = CheiaE->Text;
  for (int i = 0; i < 320; i++)   {
        str = baseR[i];
        Sleep(StrToInt(delayR->Text));
        IdTCPClientR_2->IOHandler->WriteLn(header + str + IntToStr(i));
```

```
        if(IdTCPClientR_2->IOHandler->ReadByte() == 1) {
          RFinalBases->Text = RFinalBases->Text + baseR[i];
          RFinalQbits->Text = RFinalQbits->Text + RQbitsMemo->Lines->Text.SubString(i+1,1);
          ch = RQbitsMemo->Lines->Text.SubString(i+1,1);
          if ((ch == "a") || (ch == "c"))   {
                    finalkeyR = '0';
          }
          else if (((ch == "b") || (ch == "d")))   {
                    finalkeyR = '1';
          }
          RFK->Text = RFK->Text + finalkeyR;
        } else   {
           RFinalBases->Text = RFinalBases->Text + '*';
        }
  }
  NrBitiCheiaBruta->Text =  RFK->GetTextLen();
}
//---------------------------------------------------------------------------
void __fastcall TFormReceptor::IdTCPServerR_3Execute(TIdContext  *AContext)
{
  UnicodeString line, ch, head, sir, qbit, tstamp, TStamp1, TStamp2, timp;
  Word Hour1, Min1, Sec1, MSec1, Hour2,  Min2, Sec2, MSec2;
  TDateTime time2;
  line = AContext->Connection->IOHandler->ReadLn("");
  TDateTime time1 = Now();
  head = line.SubString(1,1); // extragem "head"
  if (head == "W") { // primim Qbitii de la Inamic
        DecodeTime(time1, Hour1, Min1, Sec1, MSec1);
        TStamp1=IntToStr(Hour1)+"."+IntToStr(Min1)+"."+IntToStr(Sec1)+"."+
IntToStr(MSec1);
        time1 = EncodeTime(Hour1, Min1, Sec1, MSec1);
        sir = line.SubString(2, line.Length());
        qbit = sir.SubString(LastDelimiter(".", sir) + 1, sir.Length());
        tstamp = sir.SubString(1, LastDelimiter(".", sir) - 1);
        MSec2 = StrToInt(tstamp.SubString(LastDelimiter(".",  tstamp) +1 , tstamp.Length()));
        tstamp = tstamp.SubString(1,  LastDelimiter(".", tstamp) - 1);
        Sec2 =  StrToInt(tstamp.SubString(LastDelimiter(".",  tstamp) + 1, tstamp.Length()));
        tstamp = tstamp.SubString(1,  LastDelimiter(".", tstamp) - 1);
        Min2 = StrToInt(tstamp.SubString(LastDelimiter(".",  tstamp) + 1, tstamp.Length()));
        tstamp = tstamp.SubString(1,  LastDelimiter(".", tstamp) - 1);
        Hour2 = StrToInt(tstamp.SubString(LastDelimiter(".",  tstamp) + 1, tstamp.Length()));
        TStamp2=IntToStr(Hour2)+"."+IntToStr(Min2)+"."+IntToStr(Sec2)+"."+
IntToStr(MSec2);
        time2 = EncodeTime(Hour2, Min2, Sec2, MSec2);
        timp = FloatToStr(MilliSecondsBetween(time2,  time1));
        if (MilliSecondsBetween(time2, time1) >= 160) {
                  InamicL->Font->Color = clRed;
                  InamicL->Caption = "INAMIC PREZENT";
```

```
        } else {
                InamicL->Caption = "INAMIC ABSENT";
        };
        RFinalBases->Text = RFinalBases->Text + timp + " " + IntToStr(count) + " ";
   count++;
        RQbitsMemo->Text = RQbitsMemo->Text + qbit;
 } else if (head == "A") {
                UnicodeString CF, TC, TT="";
                int nr1, nr2, nr3;
                char ch;
                CF = CheiaFinala->Text;
                TC = line.SubString(2,line.Length());
                for (int i = 1; i <= TC.Length(); i=i+3) {
                  nr1 = StrToInt(TC.SubString(i,3));
                  nr2 = StrToInt(CF.SubString(i,3));
                  nr3 = nr1 - nr2;
                  ch = (char) nr3;
                  TT += ch;
                }
                TextClar->Text = TT;
        }
}
//---------------------------------------------------------------------------
void __fastcall TFormReceptor::IdTCPServerR_3Connect(TIdContext  *AContext)
{
        if(Label1->Caption == "NECONECTAT")
                if(AContext->Connection->IOHandler->Connected())        {
                  Label1->Caption = "CONECTAT";
                  ImgC->Visible = True;
                  ImgN->Visible = False;
                  Image4->Visible = False;
                  Image2->Visible = True;
                }
}
//---------------------------------------------------------------------------
void __fastcall TFormReceptor::IdTCPServerR_3Disconnect(TIdContext  *AContext)
{
        if(Label1->Caption == "CONECTAT")
                if(!(AContext->Connection->IOHandler->Connected())) {
                  Label1->Caption = "NECONECTAT";
                  ImgC->Visible = False;
                  ImgN->Visible = True;
                  Image2->Visible = False;
                  Image4->Visible = True;
                }
}
//---------------------------------------------------------------------------
void __fastcall TFormReceptor::SKRClick(TObject *Sender)
```

```
{
  if(!IdTCPClientR_2->Connected()) {
          ShowMessage("Conexiune nerealizata !!");
          return;
  }
  UnicodeString line;
  UnicodeString header = "K";
  IdTCPClientR_2->IOHandler->WriteLn(header + RFK->Text);
  line = IdTCPClientR_2->IOHandler->ReadLn("");
  CheiaE->Text = line;
  for (int i = 0; i <= CheiaE->GetTextLen(); i++)  {
         if (CheiaE->Text.SubString(i+1,1) == RFK->Text.SubString(i+1,1))    {
           CheiaFinala->Text = CheiaFinala->Text + RFK->Text.SubString(i+1,1);
         }
  }
  NrBitiCheiaFinala->Text = CheiaFinala->GetTextLen();
}
//---------------------------------------------------------------------------
```

Inamic.cpp

```
//---------------------------------------------------------------------------
#include <vcl.h>
#pragma hdrstop
#include "Inamic.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma link "IdCustomTCPServer"
#pragma link "IdTCPServer"
#pragma link "IdCustomTCPServer"
#pragma link "IdTCPServer"
#pragma link "IdContext"
#pragma link "IdSimpleServer"
#pragma resource "*.dfm"
TFormInamic *FormInamic;
//---------------------------------------------------------------------------
__fastcall TFormInamic::TFormInamic(TComponent*  Owner) : TForm(Owner)
{
}
//---------------------------------------------------------------------------
char baseE[320] = "";
char qbitsI[320] = "";
char finalkeyI;
char baseI[320] = "";
char IqbitsR[320];
UnicodeString CheiaI;
UnicodeString baza, qbit;
```

```
void __fastcall TFormInamic::FormShow(TObject *Sender)
{
  randomize();
  for (int i = 0; i < 320; i++) {
        switch (random(2)) {
              case 0 :
                    baseI[i] = 'L';
                    break;
              case 1 :
                    baseI[i] = 'D';
                    break;
        }
  }
  IBasesMemo->Text = baseI;
  NrQbitiI->Text = "0";
  NrQbitiE->Text = "0";
}
//---------------------------------------------------------------------------
void __fastcall TFormInamic::IdTCPServerI_1Execute(TIdContext *AContext)
{
  UnicodeString line, qbit, head, sir, tstamp;
  int pos, val;
  line = AContext->Connection->IOHandler->ReadLn("");
  head = line.SubString(1,1);
  UnicodeString headW = "W";
  if (head == "Q")  {
        sir = line.SubString(2, line.Length());
        qbit = sir.SubString(LastDelimiter(".", sir) + 1, sir.Length());
        tstamp = sir.SubString(1, LastDelimiter(".", sir) - 1);
        if (CheckBox1->Checked == True) {
          IQbitsMemo->Text = IQbitsMemo->Text + qbit;
          Sleep(50);
        };
        Sleep(10);
        IdTCPClientI_3->Socket->WriteLn(headW + sir);
  }
  else if (head == "T") {
        val = StrToInt(line.SubString(2, 1));
        pos = StrToInt(line.SubString(3, line.Length()));
        if((pos >= 0) && (pos < 320)) {
          if (val == 1)     {
              qbit = IQbitsMemo->Text.SubString(pos, 1);
              if ((qbit == "a") || (qbit == "c")) {
                    CheiaI = CheiaI + "0";
              }
              if ((qbit == "b") || (qbit == "d")) {
                    CheiaI = CheiaI + "1";
              }
```

```
          }
        }
 }
 else if ((head == "C")) {
        UnicodeString CF, TC, TT="";
        int nr1, nr2, nr3;
        char ch;
        CF = IFK->Text; // cheia bruta
        TC = line.SubString(2,line.Length());
        for (int i = 1; i <= TC.Length(); i=i+3) {
          nr1 = StrToInt(TC.SubString(i,3));
          nr2 = StrToInt(CF.SubString(i,3));
          nr3 = nr1 - nr2;
          ch = (char) nr3;
          TT += ch;
        }
        TextCriptat->Text = TC;
        TextDecriptat->Text = TT;
        UnicodeString headA = "A";
        if(!IdTCPClientI_3->Connected()) {
          ShowMessage("Conexiune nerealizata !!");
          return;
        }
        IdTCPClientI_3->Socket->WriteLn(headA + TextCriptat->Text);
 }
 IFK->Text = CheiaI;
 NrBitiFK->Text = IFK->GetTextLen();
}
//---------------------------------------------------------------------------
void __fastcall TFormInamic::IdTCPServerI_1Connect(TIdContext  *AContext)
{
        if(Label6->Caption == "NECONECTAT")
                if(AContext->Connection->IOHandler->Connected())        {
                        Label6->Caption = "CONECTAT";
                        ImgCR->Visible = True;
                        ImgNR->Visible = False;               }
}
//---------------------------------------------------------------------------
void __fastcall TFormInamic::IdTCPServerI_1Disconnect(TIdContext  *AContext)
{
        if(Label6->Caption == "CONECTAT")
                if(!(AContext->Connection->IOHandler->Connected())) {
                        Label6->Caption = "NECONECTAT";
                        ImgCR->Visible = False;
                        ImgNR->Visible = True;
                }
}
//---------------------------------------------------------------------------
```

```
void __fastcall TFormInamic::ButtonConectareIClick(TObject *Sender)
{
 try     {
         if(ButtonConectareI->Caption == "Conecteaza Receptor")  {
                 IdTCPClientI_3->Host = IEditHostR->Text;
                 IdTCPClientI_3->Port = StrToInt(IEditPortR->Text);
                 IdTCPClientI_3->Connect();
                 if(IdTCPClientI_3->Connected())       {
                         ButtonConectareI->Caption = "Deconecteaza Receptor";
                         ImgCE->Visible = True;
                         ImgNE->Visible = False;
                         Label11->Caption = "CONECTAT";
                 }
return;
         }
         if(ButtonConectareI->Caption == "Deconecteaza Receptor") {
                 if(IdTCPClientI_3->Connected())
                         IdTCPClientI_3->Disconnect();
                 if(!IdTCPClientI_3->Connected()) {
                         ButtonConectareI->Caption = "Conecteaza Receptor";
                         ImgCE->Visible = False;
                         ImgNE->Visible = True;
                         Label11->Caption = "NECONECTAT";
                 }
         return;
         }
         }
         catch (EIdException &e)       {
                 ShowMessage(e.Message);
         }
}
//---------------------------------------------------------------------------
void __fastcall TFormInamic::ButtonTrimiteClick(TObject *Sender)
{
  UnicodeString headW = "W";
  if(!IdTCPClientI_3->Connected()) {
            ShowMessage("Conexiune nerealizata !!");
            return;
  }
  IdTCPClientI_3->Socket->WriteLn(headW + IQbitsInamic->Text);
}
//---------------------------------------------------------------------------
```