Universitatea "Alexandru Ioan Cuza" Iași

Facultatea de Informatică

LUCRARE DE LICENȚĂ

Protocoale de Signcriptare

propusă de

Dănuț-Petrică Andrișan

Sesiunea: Iulie, 2017

Coordonator științific

Lector, Dr. Sorin Iftene

Universitatea "Alexandru Ioan Cuza" Iași

Facultatea de Informatică

Protocoale de Signcriptare

Dănuț-Petrică Andrișan

Sesiunea: Iulie, 2017

Coordonator științific

Lector, Dr. Sorin Iftene

DECLARAȚIE PRIVIND ORIGINALITATE ȘI RESPECTAREA DREPTURILOR DE AUTOR

Prin prezenta declar că Lucrarea de licenţă cu titlul „Protocoale de Signcriptare" este scrisă de mine şi nu a mai fost prezentată niciodată la o altă facultate sau instituţie de învăţământ superior din ţară sau străinătate. De asemenea, declar că toate sursele utilizate, inclusiv cele preluate de pe Internet, sunt indicate în lucrare, cu respectarea regulilor de evitare a plagiatului:

- toate fragmentele de text reproduse exact, chiar şi în traducere proprie din altă limbă, sunt scrise între ghilimele şi deţin referinţa precisă a sursei;
- reformularea în cuvinte proprii a textelor scrise de către alţi autori deţine referinţa precisă;
- codul sursă, imaginile etc. preluate din proiecte open-source sau alte surse sunt utilizate cu respectarea drepturilor de autor şi deţin referinţe precise;
- rezumarea ideilor altor autori precizează referinţa precisă la textul original.

Iași, 24.06.2017

Absolvent Dănuț-Petrică Andrișan

DECLARAŢIE DE CONSIMŢĂMÂNT

Prin prezenta declar că sunt de acord ca Lucrarea de licență cu titlul „Protocoale de Signcriptare", codul sursă al programelor şi celelalte conţinuturi (grafice, multimedia, date de test etc.) care însoţesc această lucrare să fie utilizate în cadrul Facultăţii de Informatică. De asemenea, sunt de acord ca Facultatea de Informatică de la Universitatea „Alexandru Ioan Cuza" Iași să utilizeze, modifice, reproducă şi să distribuie în scopuri necomerciale programele-calculator, format executabil şi sursă, realizate de mine în cadrul prezentei lucrări de licenţă.

Iași, 24.06.2017

Absolvent Dănuț-Petrică Andrișan

Cuprins

Capitolul 1

Introducere

O dată cu dezvoltarea internetului informația transmisă prin internet a început să aibă nevoie de o anumită protecție. Transmisia de informaţii senzitive, spre exemplu informaţiile privind cardurile de credit sau tranzacţile bancare reuşesc să atragă mulţi admiratori cu intenţie rea. Astfel, informaţia a devenit valoroasă, şi ca orice obiect valoros trebuie protejată.

Securitatea informaţiei se ocupă cu protejarea informaţilor şi a sistemelor informatice împotriva acţiunilor neautorizate în ceea ce priveşte accesul, folosirea, copierea, modificarea sau distrugerea datelor.

Criptografia este unul din instrumentele folosite în securitatea informaţilor. Este stiinţa ce reuşeste să ascundă informaţia de restul lumii, chiar dacă este text, audio sau imagine, ceea ce face dificilă aflarea informaţiei pentru entităţile nedorite. Mecanismele sale de bază sunt criptarea, care reuşeste să asigure confidenţialitate datelor, şi semnarea digitală ce asigură autenticitatea originii și integritatea datelor.

Fie următorul scenariu în care un doctor vrea să trimită informații medicale despre un pacient către baza de date medicală. În astfel de scenariu:

(1) vrem să asigurăm faptul că înregistrările medicale ale pacientului rămân *confidențiale*

(2) vrem să asigurăm și *autenticitatea* acestora pentru a fi siguri că persoana care transmite informațiile este într-adevăr medicul pacientului și aceste informații nu au fost modificate

O soluție la problema anterioară este metoda clasică *"semnează și apoi criptează"*, find o combinare a primitivelor de criptare și semnare digitală. Astfel această metodă adaptată la exemplu curent, necesită ca doctorul să semneze mesajul pentru a asigura autenticitatea, după care va trebui să cripteze mesajul și semnătura anterioară pentru asigurarea confidențialității datelor. Dar, acest mecanism s-a dovedit ineficient, datorită costului computaţional şi de comunicare mare, astfel o întrebare importantă a luat naștere:

Este posibil de-a trimite un mesaj de lungime arbitrară cu cost computaţional şi de comunicare mai mic decât metoda "semnează și apoi criptează" ?

În anul 1997 Yuliang Zheng a venit cu un răspuns pozitiv la această întrebare descoperând o nouă primitivă criptografică, numită signcriptarea.

Signcriptarea este primitiva criptografică în criptografia cu chei publice ce reuşeste să îndeplinească simultan funcţiile de semnătură digitală şi criptare însă într-un mod mult mai eficient decât combinare clasică "semnează și apoi criptează".

În această lucrare vom descrie progresul actual a acestei topici, prezentând principale protocoale care au contribuit la dezvoltarea acestei primitive, care a avut o evoluție optimistă în privința securității, reușind să descrie unele protocoale care satisfac majoritatea proprietăților de securitate necesare acestor scheme. Deasemenea parcurgem cele mai importante și cunoscute variații a acestei primitive în care vom vedea cum sunt folosite aceste protocoale în aplicații, ca plata electronică, agenți mobili sau rețele mobile ad-hoc.

Lucrarea este organizată în 7 capitole. Primul capitol este unul introductiv în care sunt prezentate noțiuni introductive despre signcriptare, al doilea capitol este dedicat primitivelor criptografice ce vor fi utilizate în protocoalele prezentate, în capitolul doi vom prezenta protocolul de bază de signcriptare, capitol trei v-a fi dedicat evoluției acestei primitive în criptografia bazată pe identitate, al patrulea capitol cuprinde cele mai cunoscute și importante variații ale primitivei, în capitolul cinci este descrisă aplicația și detalii privind implementarea acesteia, iar în ultimul capitol sunt prezentate o serie de concluzii și direcții viitoare de cercetare.

Capitolul 2

Primitive criptografice

În acest capitol vom prezenta principalele primitive criptografice necesare introducerii schemelor de signcriptare. Este vorba de primitivele simetrice (criptosistem, funcții hash), primitive asimetrice (criptosistem, semnătura digitală), primitivele bazate pe identitate şi schemele de partajare a secretelor, preluate în principal, din [24].

2.1 Criptografia simetrică

Criptografia simetrică se referă mai ales la metode de criptare/decriptare în care atât transmiţătorul cât şi receptorul folosesc aceeaşi cheie. Aceasta trebuie stabilită înainte de părţile implicate. Astfel se formează o pereche de algoritmi determiniști (E, D) în care:

- algoritmul de criptare E primește o cheie simetrică K și un mesaj $m \in \mathcal{M}$ și va returna un criptotext $C \in \mathcal{C}$, $C \leftarrow E_K(m)$
- algoritmul de decriptare D primește cheia simetrică K și criptotextul $C \in \mathcal{C}$ și va returna $m \in \mathcal{M}$ în caz de succes sau simbolul eroare $\perp$, $m \leftarrow D_K(C)$.

Din clasa primitivelor criptografice simetrice fac parte şi funcţiile hash (cu cheie).O *funcţie hash* este o funcţie ce transformă stringuri de lungimi arbitrare, în stringuri de lungime fixă, numite rezumate hash. Funcţiile hash trebuie construite astfel încât pentru două şiruri de intrare să nu rezulte acelaş rezumat, aceasta

find proprietatea de rezistenţă la coliziune. Deasemenea o funcţie hash one-way se caracterizează prin dificultatea de a recreea din rezumatul unei funcţii hash, şirul de intrare iniţial folosit.

2.2 Criptografia asimetrică

Criptografia cu chei publice, sau asimetrică, se caracterizează prin folosirea perechilor de chei: o cheie publică pentru criptarea datelor, şi o cheie privată sau secretă pentru decriptare. Cheia publică este accesibilă pentru toţi uilizatori, iar cheia privată este păstrată secret. Astfel, acest criptosistem este format din trei algoritmi (Gen, E, D) în care:

- algoritmul de generare Gen este un algoritm probabilistic care primește un parametru de securitate 1^k și va returna o pereche de chei $(x_{Gen}, y_{Gen}) \leftarrow KeyGen(1^k)$. Cheia publică x_{Gen} este distribuită tuturor utilizatorilor în timp ce cheia privată y_{Gen} este păstrată secretă.
- algoritmul de criptare E este un algoritm probabilistic care primește un mesaj $m \in \mathcal{M}$ și cheia publică y_{Gen} și va returna criptotextul $C \in \mathcal{C}$, $C \leftarrow E(y_{Gen}, m)$
- algoritmul de decriptare D este un algoritm determinist care primește criptotextul $C \in \mathcal{C}$ și cheia privată x_{Gen} și va returna mesajul $m \in \mathcal{M}$ în caz de succes sau simbolul eroare $\perp$, $m \leftarrow D(x_{Gen}, C)$

Un beneficiu major al criptografiei cu chei publice este furnizarea unei metode pentru realizarea *schemelor de semnare digitală*. *Semnăturile digitale* permit unui receptor să verifice autenticitatea şi integritatea informaţii primite. Deasemenea semnăturile digitale asigură non-repudierea, prin care transmiţătorul nu poate nega faptul că el a transmis informaţia. Semnătura digitală implică două procese: crearea semnăturii şi verificarea semnăturii. În generarea semnăturii se foloseşte cheia privată iar în verificarea semnăturii se foloseşte cheia publică, care corespunde cheii private. Astfel un mesaj semnat se poate verifica de orice utilizator care cunoaşte cheia publică corespunzătoare chei private de semnare a mesajului.

2.3 Criptografia bazată pe identitate

Criptografia bazată pe identitate se caracterizează prin abilitatea de a folosi orice string ce identifică unic un utilizator(de exemplu, adresa de e-mail), ca cheie publică. În astfel de model este necesară existenţa unei autorităţi de încredere numită Private Key Generator(PKG), a cărui sarcină este de a genera cheia privată a unui utilizator folosind informaţii despre identitatea sa. Un astfel de criptosistem este prezentat în [14] acesta find format din următorii algoritmi:

- **Setup :** algoritmul este rulat de PKG care va genera cheia master K împreună cu parametri sistemului. Cheia master este ținută secretă și va fi folosită pentru a crea cheile private a utilizatorilor.
- **Extragere :** algoritmul va fi rulat de PKG atunci când un utilizator necesită cheia privată. Folosind parametri sistemului, cheia master și identitatea utilizatorului va returna cheia privată x_{ID} a utilizatorului cu identitatea ID.
- **Criptarea :** algoritmul care primește parametri sistemului, un mesaj $m \in \mathcal{M}$ și o identitate ID și va returna criptotextul $C \in \mathcal{C}$.
- **Decriptarea :** algoritmul primește cheia privată x_{ID}, parametri sistemului și criptotextul $C \in \mathcal{C}$ și va returna mesajul $m \in \mathcal{M}$.

2.4 Partajarea secretelor

Partajarea secretelor este o metodă prin care se distribuie un secret între un grup de participanţi, fiecare primind o parte din secret. Secretul poate fi reconstruit doar dacă anumite părţi sunt combinate împreună, şi niciun participant a grupului nu poate dezvălui informaţii asupra secretului. În general, într-o astfel de schemă există un arbitru (un dispozitiv de încredere) care are un secret, şi n participanţi care au părţile generate din secret. Arbitrul va trebui să distribuie fiecărui participant o parte din secret în aşa fel încât cel puţin un număr fix(prag) de participanţi, t, să poată reconstrui secretul, şi nici o informaţie să nu se poată descoperi despre secret dacă numărul de participanţi este mai mic decât t. O astfel de schemă se numeşte (t, n)-prag şi a fost prezentată prima dată de Shamir în 1979. Idea principală a schemei lui Shamir este următoarea:

• Să presupunem că folosim (t,n)-prag pentru a distribui un secret S
• Alegem un număr prim p, care este o valoare publică, (p>s)
• Fie $a_0 = S$ secretul, şi t numărul de participanţi necesari pentru a reconstrui secretul unde $0< t \leq n < p$
• Se generează aleator $t-1$ valori, a_1, a_2 ..., a_{t-1}
• Se construieşte o funcţie polinomială $f(x) = a_0 + a_1x + \cdots + a_{t-1}x^{t-1}$
• Se vor construi n părți(subsecrete) cu ajutorul funcţii polinomiale
$(i, f(i))$ cu $i \neq 0$
• Se va distribui fiecărui participant o pereche de forma $(x, f(x))$, pentru valori x distincte două câte două.
Această metodă poate fi distribuită până la $n = p-1$ participanţi. Acum fiecare participant are o parte distinctă din secret. Prin urmare, S poate fi construit folosind t astfel de perechi folosind interpolarea polinomiala în forma Lagrange.

2.5 Shortened Digital Signature Schemes (SDSS)

Parametri schemei sunt:
p : un număr prim mare(1024 biți)
q : factor prim al lui p-1(160 biți)
g : un întreg în intervalul [1, ...,p-1],de ordin q modulo p
$hash$: funcție hash one-way
x : valoare generată aleator în intervalul [1, ...,p-1]
x_a : cheia privată generată aleator din [1, ..., p-1]
y_a=g^{x_a}mod p : cheia publică
Shortened Digital Signature Schemes are două versiuni: SDSS1 și SDSS2. Semnătura pe un mesaj m este compusă din două numere r și s, unde :

- $r = hash(g^x mod p)$
- $s = x/(r + x_a) \bmod q$

în cazul SDSS1 și

- $r = hash(g^x mod p)$
- $s = x/(1 + x_a * r) \bmod q$

în cazul SDSS2. Semnătura pe mesajul m va fi (r, s). Pentru a verifica semnătura este necesar să se calculeze :
$k = (y_a * g^r)^s \bmod p$, în cazul SDSS1 și
$k = (g * {y_a}^r)^s \bmod p$, în cazul SDSS2.
După obținere k mai trebuie verificat faptul că
$hash(k, m) = r$.
SDSS1 este mai eficientă decât SDSS2 în generarea semnăturii, deoarece a doua necesită o înmulțire în plus.

Capitolul 3

Signcriptare

Apariția criptografiei cu chei publice a făcut posibil pentru oameni care nu s-au întâlnit niciodată înainte să poată comunica între ei într-un mod securizat și autentificat. Să considerăm un scenariu în lumea reală când un transmițător vrea să trimită o scrisoare confidențială într-un mod în care nu poate fi contrafăcută. Pentru aceasta o practică comună pentru transmițătorul scrisorii este ca prima dată să semneze sau să autentifice mesajul și să-l pună într-un plic și apoi să-l cifreze înainte de al trimite. Astfel pentru a se asigura securitatea informaților transmise prin internet s-au folosit aceleași principii, soluția clasică pentru a transmite un mesaj într-un mod securizat și autentificat find metoda "semnează și apoi criptează ".
Această schemă este formată din 3 pași:

(1) semnarea mesajului folosind o schemă de semnare digitală pentru asigurarea autenticității originii și integrității datelor

(2) criptarea mesajului împreună cu semnătura, folosind un algoritm de criptare cu cheie privată

(3) criptarea criptotextului de la pasul 2 folosind cheia publică a receptorului

Avantajul acestei metode este faptul că asigură autenticitate prin semnarea mesajului și confidențialitate prin criptarea mesajului. Dezavantajul metodei este costul computațional și de comunicare mare, costul find suma dintre costul semnăturii și cel a criptării. Însă, nevoia simultană atât a autenticității și confidențialității în multe aplicații, a făcut ca găsirea unei soluții mai eficiente pentru această metodă

să fie doar o chestiune de timp.

3.1 Schema Zheng

În 1997, Yuliang Zheng [1] reușește să combine semnarea digitală și criptarea într-un singur pas propunând o nouă primitivă criptografică *Signcriptarea* (engl. *Signcryption*) . Această nouă primitivă satisface întotdeauna condiția :

Cost(Signcriptare)<<Cost(Semnătură)+Cost(Criptare)

Prin această inegalitate se înțelege că o schemă de signcriptare ar trebui să fie computațional semnificativ mai eficientă decât o combinare a semnăturii cu criptarea, și să producă un criptotext mai scurt decât a combinării semnăturii cu criptarea.

Zheng a reușit crearea acestei noi primitive prin combinarea a două scheme propuse de ElGamal: o schemă de criptare cu chei publice și schema de semnătura digitală scurtată, SDSS1.

Schema de signcriptare este formată din 3 algoritmi:

- *Generare Chei*(*Gen*)- un algoritm probabilistic
- *Signcriptare*(*Sc*)-un algoritm probabilistic
- *Designcriptare*(*Dsc*)-un algoritm determinist

Acesta are următorii parametri publici:
p : un număr prim mare(1024 biți)
q : factor prim al lui p-1(160 biți)
g : un întreg în intervalul [1, ...,p-1],de ordin q modulo p
hash : funcție hash one-way
KH : funcție hash one-way cu cheie
(E,D) : algoritmi de criptare și decriptare a unui algoritm criptografic cu cheie privată

Generare chei Alice:
x_a : cheia privată generată aleator din [1, ..., q-1]
y_a=g^{x_a}mod p : cheia publică

Generare chei Bob:
x_b : cheia privată generată aleator din [1, ..., q-1]
y_b=g^{x_b} mod p : cheia publică

Algoritmul de bază de *Generare chei* folosind parametri publici, va genera cheia publică și apoi cheia privată ale lui Alice și ale lui Bob.

Algoritmul de bază de *Signcriptare* a mesajului m de către Alice funcționează astfel:

(1) Alice generează aleator o valoare x din [1, ...,q-1], calculând apoi k ce este rezultatul funcției $hash$ aplicată valorii x și a chei publice a lui Bob obținându-se un string de 128 biți $k = hash(y_b^x \ mod \ p)$. Va împărți k în două parți egale de 64 de biți obținând k_1 și k_2 ce se vor folosi în criptare și algoritmul de semnare.

(2) Va cripta mesajul m folosind schema de criptare E cu cheia k_1. Rezultatul va fi criptotextul $c = E_{k_1}(m)$

(3) Va aplica funcția hash KH pe mesajul m, ce se va folosi de algoritmul SDSS1, cu cheia k_2 obținând un string r de 128 biți $r = KH_{k_2}(m)$

(4) Va calcula o valoare $s = x{\cdot}(r + x_a)^{-1} \ mod \ q$

(5) Va da ca output un criptotext (c, r, s), ce va fi trimis lui Bob.

Algoritmul de bază de *Designcriptare* a criptotextului (c, r, s) de către Bob funcționează astfel:

(1) Va recupera k (fiind identic ca cel folosit de Alice) folosind r,s primite de la Alice, valorile publice g,p,y_a și cheia privată a sa,
$k = hash((y_a \cdot g^r)^{s \cdot x_b} \ mod \ p)$

(2) Va împărți k (care este de 128 biți) în k_1 și k_2 de 64 de biți fiecare

(3) Va decripta criptotextul c folosind cheia k_1 și algoritmul de decriptare obținând $m = D_{k_1}(c)$

(4) Va verifica autenticitatea mesajului, ca find de la Alice, dacă aplicarea funcției hash KH cu cheia k_2, are rezultatul egal cu m. Dacă este egal, va returna mesajul m, dacă nu mesajul "Reject".

În cele ce urmează, vom prezenta principalele modele de securitate folosite în cadrul analizei metodelor de signcriptare, preluate din [25].
Fie S(Alice) cel care transmite mesajul și R(Bob) cel care primește mesajul. Scopul securității acestei scheme este de a furniza atât autenticitate cât și intimitate asupra datelor comunicate. În abordarea simetrică, atât S cât și R împărtășesc aceași cheie privată, de unde rezultă că singurul model de securitate este acel modelat asupra unui străin, numit *outsider security*. În abordarea cu chei publice, S și R nu au aceași cheie privată de unde rezultă că modelul de securitate este modelat atât asupra unui străin cât și asupra unui utilizator a sistemului(S, R sau a unui străin care cunoaște cheia privată a unuia dintre părți), un astfel de model de securitate numindu-se *insider security*.

În *outsider security*, se definește securitatea împotriva celor mai puternice noțiuni de securitate a autenticității, cum ar fi UF-CMA sau sUF-CMA, și a confidențialității, IND-CCA2.
Pentru a putea defini securitatea ne vom imagina următorul adversar. Vom presupune că avem un adversar $\mathcal{A}$, care are informațiile publice (pk_S,pk_R). Deasemenea are acces la oracolele de signcriptare și designcriptare. Mai exact, poate conduce atacuri cu mesaje alese asupra lui S, prin cererea lui S să producă o signcriptare C asupra unui mesaj arbitar m. Similar, poate conduce atacuri cu criptotext ales asupra lui R, trimițând orice signcriptare C și primind înapoi mesajul m. De notat faptul că $\mathcal{A}$ nu poate rula singur un oracol datorită lipsei cheilor private.

UF-CMA/sUF-CMA

Pentru a compromite securitatea UF-CMA a schemei de signcriptare, $\mathcal{A}$ trebuie să vină cu un criptotext valid a unui mesaj m, care nu a fost cerut sa fie criptat înainte.

Definiția 3.1 (preluată din [25]) Spunem că o schemă de signcriptare este *outsider secure* în sensul UF-CMA/sUF-CMA, dacă orice atacator probabilistic în timp polinomial(PPT) $\mathcal{A}$ are o șansă neglijabilă de a avea succes în jocul

următor. Pentru a compromite securitatea sUF-CMA a schemei de signcriptare, $\mathcal{A}$ trebuie să vină cu un criptotext valid, care nu a fost returnat de S anterior.
Vom considera un joc între o entitate numită challenger și un atacator PPT, $\mathcal{A}$:
1. Challengerul generează parametri comuni, param←Setare(1^k), perechile de chei a lui S, (sk_S,pk_S) ← $GenChei_S$(param), și a lui R , (sk_R,pk_R) ← $GenChei_R$(param)
2. Atacatorul rulează $\mathcal{A}$ pe intrarea (param, pk_S, pk_R). Atacatorul poate interoga un oracol de signcriptare cu un mesaj m pentru a primi un criptotext
C ← Signcriptare(sk_S,pk_R,m). Deasemenea atacatorul poate interoga un oracol de designcriptare cu criptotextul C pentru a primi mesajul
m←Designcriptare(pk_S,sk_R,C). Atacatorul termină prin returnarea criptotextului C.

Atacatorul câștigă jocul UF-CMA dacă
1. m ← Designcriptare(pk_S,sk_R,C) satisface $m \neq \perp$, adică C este criptotextul mesajului m
2. m nu a fost trimis niciodată la un oracol de signcriptare .
Atacatorul câștigă jocul sUF-CMA dacă
1. m ← Designcriptare(pk_S,sk_R,C) satisface $m \neq \perp$
2. oracolul de signcriptare nu returnează niciodată criptotextul C.

IND-CCA2
Pentru a compromite securitatea IND-CCA2 a schemei, $\mathcal{A}$ trebuie să vină cu două mesaje de lungime egală m_0 și m_1. Unul din acestea va fi signcriptat aleator de un challenger și vor fi trimise lui $\mathcal{A}$, ce va trebui să ghicească care mesaj a fost signcriptat. Atacatorul $\mathcal{A}$, are interzisă interogarea unui oracol de designcriptare.
Vom considera un joc între un challenger și un atacator PPT, $\mathcal{A}$=($\mathcal{A}_1$,$\mathcal{A}_2$):
1. Challengerul generează parametri comuni, param←Setare(1^k), perechile de chei a lui S, (sk_S,pk_S) ← $GenChei_S$(param), și a lui R, (sk_R,pk_R) ← $GenChei_R$(param)
2. Atacatorul rulează $\mathcal{A}_1$ pe intrarea (param, pk_S, pk_R). Atacatorul poate interoga un oracol de signcriptare cu un mesaj m pentru a primi un criptotext
C ← Signcriptare(sk_S,pk_R,m). Deasemenea atacatorul poate interoga un oracol de designcriptare cu criptotextul C pentru a primi mesajul
m ← Designcriptare(pk_S,sk_R,C). Atacatorul termină cu returnarea a două mesaje de lungime egală m_0,m_1 și unele informații de stare α.
3. Challengerul alege $b \leftarrow \{0, 1\}$ și calculează criptotextul
C^* ← Signcriptare(sk_S,pk_R,m_b)

4. Atacatorul rulează $\mathcal{A}_2$ pe intrarea criptotextului oponentului, C^*, și informațiile de stare, α. Atacatorul ar putea interoga oracolele de signcriptare și de designcriptare înainte, cu excepția de a trimite criptotextul C^* la un oracol de designcriptare. Atacatorul termină cu returnarea unui bit b'.
Atacatorul câștigă jocul dacă $b = b'$ iar avantajul atacatorului de a câștiga jocul este definit ca $\epsilon =|$ Pr[b=b'] - 1/2 |
Definiția 3.2 (preluată din [25]) Spunem că schema de signcriptare este *outsider secure* în sensul IND-CCA2 dacă pentru fiecare PPT atacator $\mathcal{A}$, acesta are o șansă neglijabilă pentru a avea succes în jocul anterior.

În *inside security*, noțiunile de securitate sunt asemănătoare ca cele de la *outside security*, cu excepția faptului că atacatorul are cheia privată a unui utilizator. În UF-CMA/sUF-CMA atacatorul are cheia privată a receptorului indicând faptul că o schemă de signcriptare previne ca un atacator să falsifice un criptotext signcriptat provenit de la un transmițător. Aceasta înseamnă că o schemă de signcriptare protejează autenticitatea mesajelor chiar dacă cheia receptorului a fost furată de un atacator. În jocul IND-CCA2, atacatorul are cheia privată a transmițătorului indicând faptul că o schemă de signcriptare previne ca un atacator să descifreze un criptotext signcriptat ce a fost produs anterior. Aceasta înseamnă că o schemă de signcriptare protejează confidențialitatea mesajelor chiar dacă cheia transmițătorului a fost furată de un atacator.

În continuare vom vorbi despre eficiență, deoarece când ne vom referi la acest gen de schemă, acest aspect este foarte important. Vom face comparația cu o metodă "semnează și apoi criptează" care este combinația dintre criptarea ElGamal și semnarea SDSS1 pe care se bazează schema de signcriptare prezentată. Când ne referim la eficiență ne referim la costul computațional și dimensiunea criptotextului de trimis. Astfel pentru costul computațional se vor lua cele mai costisitoare operații, în cazul de față find exponierea modulară. După o verificare a celor două scheme se observă că combinația dintre ElGamal și SDDS1 conține 5 exponențieri modulare dintre care 3 la criptare și decriptare și 2 la semnare și verificare, iar în cazul schemei Zhang există 2 exponențieri modulare dintre care 1 la operația de signcriptare și 1 la operația de designcriptare. Astfel o schemă de signcriptare va salva în calcul computațional și până la 60% față de o combinație ElGamal și SDSS1.

Pentru dimensiunea criptotextului se va lua în calcul numărul de biți a

variabilelor q, p și a funcților hash folosite $hash$ și KH. În combinația dintre El-Gamal și SDDS1 dimensiunea criptotextului este $|hash(*)| + |q| + |p|$ iar în cazul schemei de signcriptare este $|KH(*)| + |q|$. Astfel pentru următorul exemplu în care considerăm că $|hash(*)| = |KH(*)| = 128$, $|q| = 144$ si $|p| = 512$ se va salva în dimensiunea criptotextului până la 70%, $\frac{|hash(*)|+|q|+|p|-|KH(*)|+|q|}{|hash(*)|+|q|+|p|}$. Se poate observa faptul că, cu cât p este mai mare cu atât mai mare va fi și câștigul în dimensiune.

3.2 Schema Bao-Deng

În anumite aplicații este nevoie ca o entitate externă să verifice valabilitatea semnăturii pentru a se putea convinge că într-adevăr mesajul este trimis de transmițătorul sugerat de receptor, în cazul unei dispute asupra proveniență mesajului. În metoda tradițională "semnează și apoi criptează" mesajul și semnătură transmițătorului sunt obținute după decriptare folosind cheia privată a receptorului. Semnătura poate fi verificată folosind doar cheia publică a receptorului. Astfel, valabilitatea semnăturii poate fi verificată de oricine cunoaște cheia publică a receptorului. În schema lui Zheng pentru a putea verifica valabilitatea semnăturii este nevoie de cheia privată a receptorului. Datorită acestui fapt schema prezentată de Zheng nu asigură proprietatea de non-repudiere.

Următoarea schemă propusă de Bao și Deng în [2] reușeste să rezolve această problemă și să construiască prima schemă de signcriptare ce asigură cele trei proprietăți de securitate : *confidențialitate, autenticitate și non-repudiere.*
Schema propusă este o modificare a schemei Zheng, în care nu mai folosește pentru crearea chei de semnare, cheia publică a receptorului. Deasemenea, se înlocuiește funcția hash one-way cu cheie, KH, cu funcția hash one-way, $hash$.
Pentru descrierea schemei se vor folosi aceeași parametri ca și la schema lui Zheng și aceeași utilizatori : Alice și Bob.

Algoritmul de Signcriptare a mesajului m de către Alice funcționează astfel:
1. Alice generează aleator o valoare x din [1, ...,q-1]
2. Va calcula valoare t_1 ce va fi folosită în semnare, după cum se observă nu se mai folosește cheia publică a lui Bob, $t_1 = g^x \ mod \ p$
3. Va calcula valoare t_2 ce va fi folosită în criptare, $t_2 = y_b^x \ mod \ p$
4. Va cripta mesajul m folosind schema de criptare E cu cheia rezultată în urma

aplicării funcției $hash$ pe t_2. Rezultatul va fi criptotextul $c = E_{hash(t_2)}(m)$
5. Va aplica funcția $hash$ pe mesajul m si t_1 obținând $r = hash(m, t_1)$
6. Va calcula o valoare $s = x \cdot (r + x_a)^{-1} \ mod \ q$
7. Va returna un criptotext (c, r, s), ce va fi trimis lui Bob.

Algoritmul de Designcriptare a criptotextului (c, r, s) de către Bob funcționează astfel:
1. Va recupera valoarea t_1(fiind identic ca cel folosit de Alice) folosind r, s primite de la Alice,valorile publice g, p, y_a ca astfel: $t_1 = (y_a \cdot g^r)^s \ mod \ p$
2. Va recupera și valoarea t_2 folosindu-se de valoarea calculată anterior și cheia sa privată, $t_2 = t_1^{x_b} \ mod \ p$
3. Va decripta criptotextul c folosind cheia rezultată din aplicarea funcției $hash$ pe valoarea t_2 și algoritmul de decriptare obținând $m = D_{hash(t_2)}(c)$
4. Va verifica autenticitatea mesajului, ca find de la Alice, dacă aplicarea funcției $hash$ pe mesaj și valoare t_1, are rezultatul egal cu m. Dacă este egal, va returna mesajul m, dacă nu va returna "Reject".

Nerepudierea este acum asigurată, Bob poate trimite (m, r, s) celorlalți, care pentru a se convinge că într-adevăr mesajul provine de la Alice vor verifica dacă: $r = hash(m, (y_a g^r)^s)$

În privința securității, schema este la fel ca schema originală, asigurându-se confidențialitatea(IND-CCA2), find intractabil pentru un atacator să afle vreo informație din textul signcriptat. Deasemenea, se asigură faptul că este intractabil pentru un atacator să falsifice textul signcriptat(UF-CMA). Ceea ce aduce în plus este non-repudierea, care asigură faptul că o entitate externă poate rezolva o dispută dintre Alice și Bob în cazul în care Alice neagă faptul că ea este autorul textului signcriptat.

În privința eficienței din punct de vedere a calculului computațional, se poate observa că schema are nevoie de calcularea a 4 exponențieri modulare dintre care 2 în signcriptare și 2 în designcriptare. Astfel schema nu este la fel de eficientă ca schema lui Zheng care necesită doar 2 astfel de operații. Însă schema Bao-Deng asigură non-repudierea și este deasemenea mai eficientă decât combinarea dintre ElGamal și SDSS1 care necesită 5 exponențieri.

3.3 Schema Libert-Quisquater

Securitatea primelor scheme prezentate este bazată pe dificultatea problemei Diffie-Hellman. În acestă schemă ne vom îndrepta atenția spre aplicațiile biliniare. Datorită faptului că calcularea unei aplicații biliniare este mai înceată decât calcularea exponienților modulare, dezvoltarea unor scheme de signcriptare bazate pe aplicații biliniare au sens doar dacă aduc anumite avantaje peste schemele mult mai eficiente bazate pe Diffie-Hellman. Acestea pot fi sub forma unor proprietăți suplimentare aduse schemelor de signcriptare sau a unor noi proprietăți de securitate.

Benoît Libert și Jean-Jacques Quisquater [3] reușesc crearea unui astfel de avantaj, reușind să asigure cu schema definită anonimitatea criptotextului. Datorită acestei proprietăți aceștia reușesc să definească și o nouă noțiune de securitate numită *invisibilitatea chei*.

În prezentarea schemei vom presupune că Alice(S) și Bob(R) sunt de acord în privința parametrilor publici: k si l parametri de securitate, $\mathbb{G}_1$, $\mathbb{G}_2$ grupuri ciclice de ordin prim q, astfel încât $2^{k-1} \leq q \leq 2^k$ pentru care l este numărul de biți necesari pentru reprezentarea elementelor lui $\mathbb{G}_1$, P un generator de $\mathbb{G}_1$,
$\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ o aplicație bilineară, și trei funcții hash: $H_1 : \{0,1\}^{n+2l} \to \mathbb{G}_1$, $H_2 : \mathbb{G}_1^3 \to \{0,1\}^l$ si $H_3 : \{0,1\}^l \to \{0,1\}^{n+l}$, unde n este dimensiunea criptotextului.
Schema este formată din următorii algoritmi:

Generare chei : transmițătorul va genera aleator cheia privată, $X_S \leftarrow \mathbb{Z}_q$, și va calcula cheia publică corespunzătore chei private, $Y_S = X_S P \in \mathbb{G}_1$. La fel va genera și receptorul, perechea sa de chei, (X_R,Y_R).

Signcriptare : pentru a cripta un mesaj $m \in \{0,1\}^n$ pentru un receptor, R, transmițătorul S, va efectua urmatoarele operații:
1. Se va genera o valoarea aleatoare $r \leftarrow \mathbb{Z}_q$, și va calcula $U = rP \in \mathbb{G}_1$
2. Va calcula $V = X_S H_1(m, U, Y_R) \in \mathbb{G}_1$
3. Va calcula $W = V \otimes H_2(U, Y_R, rY_R) \in \{0,1\}^l$ și va cripta mesajul m,
$Z = (m||Y_S) \otimes H_3(V) \in \{0,1\}^{n+l}$ ($\otimes$ reprezintă operația sau pe biți)
4. Va trimite criptotextul σ=(U, W, Z) lui R.

Designcriptare : dupa primirea criptotextului σ, R va efectua următoarele operații:
1. Va parsa σ ca (U, W, Z)
2. Va calcula $V = W \otimes H_2(U, Y_R, X_R U) \in \{0,1\}^l$, pentru a putea recupera mesajul
3. Va recupera mesajul $(m||Y_S) = Z \otimes H_3(V) \in \{0,1\}^{n+l}$, și va respinge σ dacă $Y_S \notin \mathbb{G}_1$
3. Va verifica valabilitatea semnăturii calculând $H = H_1(m, U, Y_R) \in \mathbb{G}_1$ va verifica dacă $\hat{e}(Y_S, H) = \hat{e}(P, V)$, iar în cazul în care nu este adevarată se va respinge criptotextul primit.
Pentru ca o entitate externă să poată verifica faptul că S a semnat mesajul m, receptorul va trebui să trimită m și (U, V, Y_R). Pentru a fi convins de acest lucru entitatea externă va executa operațiile de la pasul 3 anterior.

Din punct de vedere a eficienței, această schemă necesită 3 operații de înmulțire în $\mathbb{G}_1$, în algoritmul de signcriptare și 1 operație de înmulțire și 2 evaluări ale aplicației bilineare, $\hat{e}$, în algoritmul de designcriptare. Eficiența acestei scheme față de combinarea clasică "criptează și apoi semnează" este datorată algoritmului de signcriptare care necesită mai puține înmulțiri.

Pe lângă proprietatea de confidențialitate(*Definiția* 3.2), schema asigură și anomitatea criptotextului(*Definiția* 3.3), garantând faptul că criptotextul ascunde atât identitatea transmițătorului cât și a receptorului, dar și proprietatea de *invisibilitate a chei* (*Definiția* 3.4) ce asigură imposibilitatea de a decide dacă criptotextul dat a fost creat cu cheia privată a transmițătorului sau cheia privată a receptorului.

Definiția 3.3 (preluată din [3]) Spunem că o schemă este *FSO/FUO-ANON-CCA-sigură* dacă oricare atacator în timp polinomial are un avantaj neglijabil în jocul următor. Jocul se desfășoară între un challenger, $\mathcal{C}$, și un adversar $\mathcal{A}$:
1. Challengerul generează două perechi de chei (sk_{R_0}, pk_{R_0}) și (sk_{R_1}, pk_{R_1}), și trimite adversarului cheile publice, pk_{R_0} și pk_{R_1}.
2. Atacatorul va executa o serie de cereri $Signcriptare(m, sk_{R_D}, pk_R)$, cu pk_R generat aleator, și $Designcriptare(\sigma, sk_{R_D})$ pentru D=0 sau D=1.
3. După terminarea pasului 2, atacatorul va returna două chei private sk_{S_0} si sk_{S_1} și un mesaj m. Challengerul va genera aleator două variabile, b,b' $\in \{0,1\}$ și va calcula un criptotext $\sigma = Signcriptare(m, sk_{S_b}, pk_{R_{b'}})$ ce va fi trimis adversarului
4. Atacatorul va executa o serie de operații adaptive ca în pasul 2 cu restricția

apelării algoritmului *Designcriptare* cu σ și cheile private sk_{R_0} și sk_{R_1}
5. La final atacatorul va returna două variabile e,$e^{'} \in \{0,1\}$ și va câștiga jocul dacă $(e,\ e^{'}) = (\ b,\ b^{'})$
Avantajul adversarului este $Adv(\mathcal{A}) = |Pr[e = b] - \frac{1}{2}|$.

Definiția 3.4 (preluată din [3]) Spunem că o schemă este *FSO/FUO-INVK-CCA-sigură* dacă oricare atacator în timp polinomial are un avantaj neglijabil în jocul următor. Jocul se desfășoară între un challenger, $\mathcal{C}$, și un adversar $\mathcal{A}$:
1. Challengerul generează o pereche de chei (sk_S, pk_S) și trimite adversarului cheia publică pk_S
2. Atacatorul va executa o serie de cereri $Signcriptare(m, sk_S, pk_R)$, cu pk_R ales aleator, și $Designcriptare(\sigma, sk_S)$
3. După terminarea pasului 2, atacatorul va returna o cheie privată sk_U și un mesaj m. Challengerul va genera aleator o variabilă, $b \in \{0,1\}$. Dacă $b = 0$, atunci va calcula un criptotext $\sigma = Signcriptare(m, sk_U, pk_S)$ ce va fi trimis adversarului. Dacă $b = 1$ challengerul va returna un criptotext σ generat aleator.
4. Atacatorul va executa o serie de operații adaptive ca în pasul 2 cu restricția apelării algoritmului de *Designcriptare* cu σ și cheia privată sk_S
5. La final atacatorul va returna o variabilă $e \in \{0,1\}$ și va câștiga jocul dacă $e = b$.

Capitolul 4

Signcriptare bazată pe identitate

O problemă a criptosistemelor cu cheie publică este dependența acestora de infrastructura cheilor publice. Pentru ca o comunicare să fie sigură trebuie să se genereze perechele de chei pentru criptare și semnare, deasemenea transmiterea certificatelor la autoritatea centrală (CA) și primirea certificatelor pentru a se autentifica. Acest proces este însă consumator de timp. Din fericire, un nou tip de criptosistem a fost propus de către Shamir în 1984 [11] pentru combaterea acestor probleme, criptosistemele bazate pe identitate(IB). Caracteristica principală a acestora constă în faptul că cheia publică poate fi orice string ce identifică unic un utilizator. Chiar dacă cheia publică poate fi obținută de oricine din informațiile publice, cheia privată corespunzătoare poate fi derivată doar de o autoritate de încredere numită Private Key Generator(PKG).

În descrierea originală, Shamir a anticipat folosirea criptografiei IB pentru semnare și criptare, însă doar semnarea IB a fost sugerată bazată pe tehnicile cunoscute la momentul acela. Au urmat o serie de descrieri a unor scheme de semnare bazate pe IB, dar o primă descriere a unei scheme de criptare IB a apărut abia în 2001 descrisă de către Boneh și Franklin [14].
Având definit o metodă de criptare, IBE, în care Alice poate trimite securizat lui Bob un mesaj, folosind ca cheie publică orice string ce-l identifică pe Bob, și deasemenea o schemă de semnare care folosește aceleași principii pentru cheia publică, IBS, a mai rămas un singur pas de făcut, combinarea celor două scheme pentru rezultarea unei noi primitive criptografice, numită *Signcriptare bazată pe identitate*.

4.1 Schema Malone

John Malone-Lee propune prima schemă de signcriptare bazată pe identitate ce a luat în considerare două noțiuni de securitate: *confidențialitatea* și $non-repudierea$ în [12].

Fie grupurile ciclice ($\mathbb{G}$,+) și ($\mathbb{V}$,$\cdot$) de ordin q, q prim. Fie P un generator pentru $\mathbb{G}$ și $\hat{e}$: $\mathbb{G} \times \mathbb{G} \to \mathbb{V}$ o aplicație bilineară ce satisface condițiile :

(1) Pentru toți P,$Q \in \mathbb{G}$ și toți a,$b \in \mathbb{Z}$ să rezulte $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$

(2) Există P,$Q \in \mathbb{G}$ astfel încât $\hat{e}(P, Q) \neq 1$.

(3) Există un algoritm eficient care poate calcula $\hat{e}(P, Q)$ pentru oricare P,$Q \in \mathbb{G}$

Deasemenea $n \in \mathbb{N}$ este lungimea mesajului ce va fi trimis și $\mathbb{G}^*$ reprezintă $\mathbb{G}\backslash\{0\}$.

Setup : dat un parametru de securitate k, PKG va alege grupurile $\mathbb{G}$, $\mathbb{V}$ de ordin prim q, generatorul P pentru $\mathbb{G}$, o aplicație bilineară
$\hat{e}$: $\mathbb{G}\times\mathbb{G} \to \mathbb{V}$ și funcțiile hash $H_1 : \{0,1\}^* \to \mathbb{G}^*$, $H_2 : \{0,1\}^* \to {\mathbb{Z}_q}^*$,
$H_3 : {\mathbb{Z}_q}^* \to \{0,1\}^n$. Va genera secretul master, $t \leftarrow {\mathbb{Z}_q}^*$, acesta find folosit pentru generarea cheilor private corespunzatoare identităților ID, a utilizatorilor care contactează PKG în acest context, și va calcula cheia publică globală $Q_{TA} = tP$, ce va fi folosită pentru generarea cheilor publice corespunzătoare unei identități ID primită.

Extragere(ID) : pentru o identitate dată ID, PKG va calcula cheia publică $Q_{ID} \leftarrow H_1(ID)$ și cheia privată $S_{ID} \leftarrow tQ_{ID}$.

Signcriptare(S_{ID_A},ID_B,m) : pentru ca Alice să trimită un mesaj m lui Bob se va folosi de cheia sa privată și identitatea lui Bob urmând pași următori:
1.Va calcula cheia publică corespunzătoare identității lui Bob, $Q_{ID_B} = H_1(ID_B)$

2.Va genera aleator $x \in \mathbb{Z}_q{}^*$, după care va calcula $U = xP$ și valoarea variabilei r prin concatenarea mesajului cu valoarea variabilei calculate anterior, $r = H_2(U||m)$
3.Va calcula $W = xQ_{TA}$ si $V = rS_{ID_A} + W$
4.Va calcula $y = \hat{e}(W, Q_{ID_B})$, $k = H_3(y)$, pentru a obține criptarea mesajului, $c = k \oplus m$
5.Va trimite criptotextul $\sigma = (c, U, V)$ lui Bob.

Designcriptare(ID_A,S_{ID_B},σ) : după ce va primi mesajul criptat Bob va efectua următorii pași:
1.Va calcula cheia publică corespunzătoare identității lui Alice, $Q_{ID_A} = H_1(ID_A)$
2.Va extrage c, U, V din criptotextul σ
3.Va calcula $y = \hat{e}(S_{ID_B}, U)$, $k = H_3(y)$ pentru a putea recupera mesajul
4.Va recupera mesajul $m = k \oplus c$
5.Va calcula $r = H_2(U||m)$ și va returna mesajul m dacă condiția următoare este adevărată: $\hat{e}(V, P) = \hat{e}(Q_{ID_A}, Q_{TA})^r \cdot \hat{e}(U, Q_{TA})$, altfel va returna $\perp$.

În vederea analizei eficienței, vom compara schema cu o metodă clasică, "criptează și apoi semnează", formată dintr-o combinație a schemei de criptare [13] și a schemei de semnare [14]. Pentru costul computațional se vor lua în calcul evaluările aplicații bilineare $\hat{e}$, operațiile exponențiale dar și cele de înmulțire. Astfel, pentru schema propusă se obține în algoritmul de *signcriptare* : 1 evaluare a lui $\hat{e}$, 3 înmulțiri în $\mathbb{G}$ iar în algoritmul de *designcriptare* : 4 evaluări ale lui $\hat{e}$, 1 operație exponențială. Verificând în același mod în metoda clasică se obține în algoritmi de *criptare/semnare* : 1 evaluare a lui $\hat{e}$, 3 înmulțiri în $\mathbb{G}$, 1 operație exponențială iar în cei de $verificare/decriptare$: 3 evaluări ale lui $\hat{e}$, 2 înmulțiri în $\mathbb{G}$.
Astfel se poate constata că schema propusă este mai eficientă decât metoda clasică, "criptează și apoi semnează", salvând o operație în generarea criptotextului și având un număr egal de operații în validarea lui.
Deasemenea criptotextele sunt mult mai compacte decât cele produse de "criptează și apoi semnează", IBSC construind criptotexte de lungime
$n + 2 \mid \mathbb{G} \mid$ față de $2n + 3 \mid \mathbb{G} \mid$, cu n numărul de biți a mesajului și $\mid \mathbb{G} \mid$ lungimea unui element $\mathbb{G}$.

Din perspectiva securităţii, următoarele proprietăţi sunt asigurate de această schemă:

• *Confidenţialitatea mesajului*(IND-ISC-CCA) : se garantează faptul că se păstrează secret mesajele schimbate între entităţi
• *Nerepudierea semnăturii*(EF-ISC-CMA) : se garantează păstrarea semnăturii pe mesaj a transmiţătorului chiar dacă se află cheia secretă a receptorului

Definiţia 4.1 (preluată din [12]) Spunem că o schemă bazată pe identitate (IBSC) este *IND-IBSC-CCA-sigură* dacă orice adversar polinomial mărginit are un avantaj neglijabil în următorul joc.
Jocul se desfășoară între un challenger, $\mathcal{C}$, și un adversar $\mathcal{A}$:

Setare : $\mathcal{C}$ va rula algoritmul de Setare pentru un parametru de securitate k dat, pentru a obține parametri publici, *param* și secretul master msk. Acesta va păstra secret msk, și va trimite *param* la adversarul $\mathcal{A}$.

Faza 1 : În timpul acestei faze adversarul va efectua o serie de cereri la $\mathcal{C}$:
-Extragere(ID_i) în care adversarul va trimite orice identitate la alegere și va primi cheia privată corespunzătoare identității.
-Signcriptare(ID_i,ID_j,m) în care adversarul va trimite două identități diferite și un mesaj, pentru a obține criptotextul mesajului m, semnat din partea primei identități și criptat pentru identitatea a doua.
-Designcriptare(ID_i,ID_j,σ) în care adversarul va trimite două identități diferite și un criptotext pentru a primi ca rezultat mesajul m sau $\perp$.
La sfârșitul fazei 1, $\mathcal{C}$ va returna două mesaje m_0 și m_1 și două identități, ID_a și ID_b, pe care se va executa provocarea.

Provocarea : $\mathcal{C}$ generează aleator b$\in\{0,1\}$ și va rula algoritmul Extragere(ID_a) și Signcriptare(S_{ID_a},ID_b,m_b). Va returna lui $\mathcal{A}$ criptotextul σ.

Faza 2 : În această fază adversarul $\mathcal{A}$ va putea face o serie de cereri ca cele descrise în Faza 1 cu următoarele restricții:
-nu sunt permise cereri de extragere pentru ID_a și ID_b
-nu este permisă cererea de designcriptare de forma Designcriptare(S_{ID_a},ID_b,m_b)

Ghicirea : $\mathcal{A}$ va returna un bit $b' \in \{0,1\}$. Acesta va câștiga doar dacă b'=b.
Avantajul adversarului este definit astfel: $\text{Adv}(\mathcal{A}) = \left|Pr[b' = b] - \frac{1}{2}\right|$.

Definiția 4.2 (preluată din [12]) Spunem că o schemă bazată pe identitate (IBSC) este EF-IBSC-CMA-sigură dacă orice adversar polinomial mărginit are un avantaj neglijabil în următorul joc.
Jocul se desfășoară astfel între un challenger, $\mathcal{C}$, și un adversar $\mathcal{A}$:

Setare : $\mathcal{C}$ va rula algoritmul de *Setare* pentru un parametru de securitate k dat, pentru a obține parametri publici, *param* și secretul master msk. Acestă va pastră secret msk, și va trimite *param* la adversarul $\mathcal{A}$.

Atac : În timpul acestei faze $\mathcal{A}$ va efectua o serie de cereri la $\mathcal{C}$, de tipul celor menționate în Faza 1 din *Definiția 4.1.*

Falsificare : $\mathcal{A}$ va returna ca rezultat două identități diferite și un criptotext, (ID_a,ID_b,σ) cu următoarele restricții:
-nu au fost efectuate cereri de extragere pentru ID_a
-nu a fost efectuată nici o cerere de signcriptare pe mesajul m
Acesta va caștiga doar dacă Designcriptare(ID_a, S_{ID_b}, σ) nu va returna $\perp$.

Avantajul adversarului este definit astfel: Adv($\mathcal{A}$) =|Pr[$\mathcal{A}$ câstigă]-$\frac{1}{2}$|.

4.2 Schema Boyen

În momentul de față comunicarea prin rețeaua de internet, este una din cele mai nesigure comunicări. Acest lucru este datorat faptului că atacatorii devin din ce în ce mai ingenioși în perspectiva unui atac. Astfel este nevoie de o comunicare cât mai sigură, acest lucru find reușit de Boyen în [7] care propune una din cele mai sigure scheme de signcriptare ce satisface majoritatea proprietăților de securitate ale schemelor de signcriptare. Schema prezentată de Boyen este formată din șase algoritmi. Deși algoritmi de *Semnare* și $Criptare$ sunt prezentați separat, aceaștia totuși constituie operația de signcriptare bazată pe identitate. Deasemenea și algoritmi de *Decriptare* și $Verificare$ definesc operația de designcriptare, dar acestia pot fi rulați și separat. Algoritmi de *Setare* și *Extragere* sunt bazați pe sistemul IBE definit de Boneh-Franklin[14].

Setup : pentru un parametru de securitate $k \in \mathbb{N}$, PKG va genera cheia master S_{ID} și cheia publică a sistemului Q_{TA}

Extragere(ID) : pentru o identitate dată ID, PKG va calcula cheia publică Q_{ID} și cheia privată corespunzatoare S_{ID}

Semnare(S_{ID_A},ID_A,m) : folosind cheia privată va semna mesajul din partea identității ID_A, și va returna, semnătura s și valoarea r

Criptare(S_{ID_A},ID_B,m,s,r) : va cripta mesajul m, pentru receptorul cu identitatea ID_B, rezultând criptotextul σ ce va fi trimis receptorului

Decriptare(S_{ID_B},$\hat{\sigma}$) : va obține identitatea transmițătorului, $\hat{ID}_A$, mesajul trimis de el, $\hat{m}$ și semnătura pe mesaj, $\hat{s}$, din decriptarea criptotextului $\hat{\sigma}$

Verificare($\hat{ID}_A$, $\hat{m}$,$\hat{s}$): va returna $\top$ (true) dacă $\hat{s}$ este o semnătură validă făcută de utilizatorul cu identitatea $\hat{ID}_A$ pe mesajul $\hat{m}$, altfel va returna $\perp$ (false).

În privința eficienței, schema propusă nu se evidențiază în vreun fel. Este mult mai eficientă decât combinația clasică "semnează și apoi criptează" și relativ asemănătoare cu schemele de signcriptare până la acel moment. Acest lucru poate fi schimbat însă dacă se renunță la proprietatea de unlinkability relativă la criptotext, acest lucru find prezentat în schema următoare.
Principala caracteristică a acestei scheme este însă securitatea pe care o asigură, putând afirma că este prima schemă de signcriptare care asigură toate proprietățile de securitate necesare schemelor de signcriptare. Pe lângă *confidențialitatea mesajului* (IND-ISC-CCA) și *non − repudierea semnăturii* (EF-ISC-CMA) prezentate în schema anterioară, Boyen reușește să atingă și următoarele proprietăți cu schema propusă:

- *Proprietatea de unlinkability relativ la criptotext*(Definiția 4.3): dă posibilitatea transmițătorului să arate că mesajul său a fost recriptat și transmis la altă entitate.
- *Autenticitatea criptotextului*(Definiția 4.4) : garantează receptorului faptul că criptarea și semnarea mesajului a fost facută de aceeași entitate. Aceasta implică de altfel și integritatea datelor.
- *Anonimitatea criptotextului*(Definiția 4.5): garantează faptul că criptotextul ascunde atât adresa transmițătorului cât și a receptorului.

Definiția 4.3 (preluată din [7]) Spunem că o schema IBSC are proprietatea de *unlinkability relativ la criptotext*, dacă există un algoritm în timp polinomial, care dacă primește un mesaj semnat (ID_A, m, s) astfel încât $Verificare(ID_A, m, s)= \top$, și o cheie privată $SK_B = Extragere(ID_B)$, să asambleze un criptotext σ, care este incontestabil computațional faptul că provine de la o criptare (m, s) făcută de ID_A pentru ID_B.

Această proprietate îi dă posibilitatea lui Alice să contrazică faptul că a trimis un anumit criptotext lui Bob, chiar dacă acel criptotext conține un mesaj sub semnătura lui Alice. Astfel spus semnătura asigură doar protecția mesajului, dar nu și a criptotextului.

Definiția 4.4 (preluată din [7]) Spunem că o schema IBSC, este *AUTH-IBSC-CMA-sigură*, dacă oricare adversar în timp polinomial are un avantaj neglijabil în jocul următor. Astfel spus, avantajul $Adv_A(k)$ = Pr[Verificare(ID_A], $\hat{m}$, $\hat{s}$)= $\top$], a oricărui adversar $\mathcal{A}$ de tip *EUF-IBSC-CMA* în timp polinomial este o funcție neglijabilă a parametrului k.
Jocul se desfășoară între challengerul $\mathcal{C}$ și adversarul $\mathcal{A}$:

Start : $\mathcal{C}$ va rula altgoritmul *Setup* pentru un parametru de securitate k dat, pentru a obține parametri publici, *param* și secretul master msk. Acesta va păstra secret msk, și va trimite *param* la adversarul $\mathcal{A}$.

Atac : În timpul acestei faze, $\mathcal{A}$ va efectua o serie de cereri la $\mathcal{C}$, de tipul celor menționate în *Definiția 4.1* .

Falsificare : $\mathcal{A}$ va returna identitatea unui receptor și criptotextul σ

Rezultat : Adversarul câștigă jocul dacă σ se poate decripta, folosind cheia privată a lui ID_B și mesajul semnat (ID_A, $\hat{m}$, $\hat{s}$) astfel încât $ID_A \neq ID_B$ satisface $Verificare(ID_S, \hat{m}, \hat{s}) = \top$ cu restricțiile:
-nu s-au mai făcut cereri de extragere a lui ID_A sau ID_B
-nu s-a făcut cerere de signcriptare pentru a calcula σ

Această proprietate dă voie receptorului să certifice faptul că criptotextul e de

la Alice dacă și mesajul e semnat de Alice, doar că nu poate demonstra acest lucru altcuiva.

Definiția 4.5 (preluată din [7]) Spunem că o schema IBSC, este *ANON-IBSC-CCA-sigură*, dacă orice adversar aleator în timp polinomial are un avantaj neglijabil în jocul următor. Astfel spus , avantajul $Adv_A = \left| Pr[\hat{b} = b'] - \frac{1}{4} \right|$ a oricărui adversar $\mathcal{A}$ de tip ANON-IBSC-CCA în timp polinomial este o funcție neglijabilă în raport cu parametrul k.
Jocul se desfășoară între challengerul $\mathcal{C}$ și adversarul $\mathcal{A}$:

Start : $\mathcal{C}$ va rula algorimul de *Setare* pentru un parametru de securitate k dat, pentru a obține parametri publici, *param* și secretul master msk. Acesta va păstra secret msk, și va trimite *param* la adversarul $\mathcal{A}$.

Faza 1 : În timpul acestei faze adversarul va efectua o serie de cereri la $\mathcal{C}$ ca cele prezentate în *Definiția 4.1*

Selecția : La un anumit punct, adversarul returnează un mesaj m, două identități a transmițătorului ID_{A_0}, ID_{A_1} și două identități a receptorului ID_{B_0}, ID_{B_1} pe care dorește să fie provocat fără ca în prealabil să fi făcut extracție pentru ID_{B_1} sau ID_{B_2}

Provocarea : $\mathcal{C}$ va alege aleatoriu b',$b''$$\in\{0,1\}$, și va rula algoritmul de *Extragere* și apoi de *Signcriptare* pentru a obține criptotextul σ ce va trimis adversarului

Faza 2 : În această fază adversarul $\mathcal{A}$ va putea face o serie de cereri ca cele descrise în *Definiția 4.1* cu următoarele restricții:
-nu sunt permise cereri de extragere pentru ID_{B_0} și ID_{B_1}
-nu este permisă cererea *Designcriptare* pentru criptotextul σ

Răspuns : $\mathcal{A}$ va returna $\hat{b}'$,$\hat{b}''$$\in \{0,1\}$ și va câștiga jocul doar dacă $(\hat{b}'$,$\hat{b}'')$ = $(b'$,$b'')$.

Această proprietate previne aflarea informațiilor despre identitațile lui Alice sau Bob din criptotext pentru cei care nu au cheia de decriptare a lui Bob. Această proprietate este adevarată chiar dacă cheia de semnare a lui Alice este expusă.

4.3 Schema Barreto-Libert bazată pe aplicații biliniare

În anumite aplicații, trebuie realizat un compromis între nivelul de securitate și eficiența schemei. În [8] se face exact acest lucru, propunându-se una din cele mai eficiente scheme, dar care nu asigură proprietatea de unlinkability relativ la criptotext. Acest lucru este datorat faptului că schema de signcriptare propusă, are la bază o schemă de semnare IBS, propusă de ei, care este mai eficientă decât toate prezentate până la acel moment.

Schema IBS propusă de ei este următoarea:
Setare: dat un parametru de securitate k, PKG alege grupurile $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ de ordin p prim, cu $p > 2^k$ și generatori $Q \in \mathbb{G}_2$, $P = \psi(Q) \in \mathbb{G}_1$, $g = \hat{e}(P, Q)$ cu aplicația bilineară $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. Se selectează cheia master $s \leftarrow {\mathbb{Z}_p}^*$, și o cheie publică la nivel de sistem $Q_{pub} = sQ \in \mathbb{G}_2$ și funcțile hash
$H_1 : \{0,1\}^* \to {\mathbb{Z}_p}^*, H_2 : \{0,1\}^* \times \mathbb{G}_T \to {\mathbb{Z}_p}^*$. Parametri publici sunt:
$\{\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, P, Q, g, Q_{pub}, e, \phi, H_1, H_2\}$

Generare Chei: pentru o identitate ID, PKG va folosi identitatea și cheia master pentru a calcula cheia privată, $S_{ID} = \frac{1}{H_1(ID)+s} P$

Semnare: pentru semnarea unui mesaj $M \in \{0,1\}$ se efectuează următoarele operații:
1. generare $x \leftarrow {\mathbb{Z}_p}^*$, și calculare $r = g^x$
2. setare $h = H_2(M, r) \in {\mathbb{Z}_p}^*$
3. calculare $S = (x + h) S_{ID}$

Verificare: semnătură $\sigma = (h, S)$ pe un mesaj M este acceptată dacă
$h = H_2\left(M, \hat{e}\left(S, H_1(ID) Q + Q_{pub}\right) g^{-h}\right)$

Eficiența schemei constă în folosirea unei singure evaluări a aplicații biliniare, $\hat{e}$, în algoritmul de verificare. În comparație cu alte scheme IBS, schema de mai sus este de aproape două ori mai rapidă la verificare și într-o măsură semnificativă la semnare.

În privința securității, schema este EUF-IBS-CMA-sigură dacă orice adversar probabilistic în timp polinomial are un avantaj neglijabil în jocul de mai jos:

(1) Challengerul generează parametri sistemului și îi trimite adversarului

(2) Adversarul va performa o serie de cereri la următoarele oracole:

- *Extragere* : va returna cheia privată a unei identități
- *Semnare* : va semna un mesaj cu o cheie privată ce aparține unei identități

(3) Adversarul va produce o tripletă (ID^*, M^*, σ^*) făcută de o identitate ID^* a cărei chei private nu a fost cerută, și deasemenea o pereche mesaj-semnătură (M^*,σ^*), pe care nu s-a făcut o cerere de semnare. Adversarul va câștiga dacă algoritmul de verificare va accepta tripleta (ID^*, M^*, σ^*).

Schema IBSC propusă este o combinație dintre schema IBS, prezentată anterior, și o versiune IBE propusă de Sakai-Kasahara[15].

Setup: : se vor folosi aceeași parametri ca și la algoritmul de *Setup* prezentat în schema anterioară la care se adaugă un nou parametru public, o funcție hash folosită la signcriptare $H_3 : G_T \rightarrow \{0,1\}^n$

Keygen: pentru o identitate ID, PKG va folosi identitatea și cheia master pentru a calcula cheia privată, $S_{ID} = \frac{1}{H_1(ID)+s}Q$

Signcriptare: pentru o identitate a transmițătorului ID_A și o identitate a receptorului ID_B și un mesaj $M \in \{0,1\}$ se vor urma pași urmatori:
1. Se va genera $x \leftarrow \mathbb{Z}_p{}^*$, și se va calcula $r = g^x$
2. Se va cripta mesajul $c = M \oplus H_3(r) \in \{0,1\}^n$
3. Se va seta $h = H_2(M, r) \in \mathbb{Z}_p{}^*$
4. Se va calcula $S = (x + h)\psi(S_{ID_A})$
5. Se va calcula $T = x(H_1(ID_B)P + \psi(Q_{pub}))$
6. Se va trimite criptotextul $\sigma = (c, S, T) \in \{0,1\} \times G_1 \times G_1$

Designcriptare: dat un criptotext $\sigma = (c, S, T)$ și identitatea ID_A se vor urma pași următori pentru a obține mesajul și a verifica semnătura:
1. Se va calcula $r = (T, S_{ID_B})$,

2. Se va recupera mesajul $M = c \oplus H_3(\mathrm{r})$
3. Se va calcula $h = H_2(M, r)$ pentru verificarea semnăturii
4. Se va returna mesajul M daca condiția $r = e(S,\ H_1(ID_A)Q +\ Q_{pub})G^{-h}$ este satisfăcută și semnătura $(h, S) \in \mathbb{Z}_p{}^* \times G_1$

Eficiența acestei scheme constă în faptul că nu trebuie să se evalueze funcții hash în grupuri ciclice. După cum Smart și Vercauteren explică în articolul lor[26], este greu de găsit grupuri G_2 pentru care să se ofere eficiență pentru stringurile aleatoare ce vor fi folosite în funcțiile hash. Astfel, schema propusă este și de 2 până la 3 ori mai rapidă la algoritmul de signcriptare și într-o măsură semnificativă la designcriptare față de cele mai bune scheme din acel timp.
Schema este $IND\text{-}IBSC\text{-}CCA$-sigură și $ESUF\text{-}IBSC\text{-}CMA$-sigură și deasemenea asigură *anonimitatea criptotextului* și *autentificarea criptotextului* sub aceleași presupuneri ca ale schemei lui Boyen[7]. Chiar dacă nu asigură proprietatea de unlinkability relativ la criptotext, aceasta este foarte importantă datorită eficienței.

4.4 Signcriptarea certificateless

Criptografia bazată pe identitate a rezolvat problema criptografiei cu chei publice reușind să combată necesitatea unei autorități de certificare a cheilor publice, însă nu a rezolvat și necesitatea autorității de încredere PKG(Private Key Generator) care autentifică utilizatori și generează cheile private care sunt transmise printr-un canal securizat utilizatorilor. Problema criptografiei bazate pe identitate se rezumă la faptul că cheia privată a fiecărui utilizator a sistemului este cunoscută de PKG care poate cripta și decripta mesaje sau falsifica semnătura oricărui utilizator. În 2003, Al-Riyani și Peterson prezintă în [28] pentru prima dată noțiunea de criptografie certificateless. Aceasta este caracterizată de două proprietăți:

(1) schema asigură securitate fără a fi nevoie ca cheia publică să fie verificată de un certificat digital

(2) schema rămâne sigură asupra atacurilor făcute de oricare terță parte (incluzând un utilizator a sistemului sau chiar PKG)

Astfel pentru ca schema să fie sigură împotriva PKG procesul de generare a cheilor este împărțită între PKG și utilizator. În această abordare PKG generează cheia privată parțială a unui utilizator, pe care utilizatorul o combină cu o valoare secretă (cunoscută doar de el) pentru a-și genera cheia privată. În consecință PKG nu mai cunoaște cheile private a utilizatorilor. Signcriptarea certificateless a fost prezentată pentru prima dată în [4], de către Barbosa și Farshim. Schema prezentată este formată din următorii șapte altgoritmi:

(1) **Setup**(k): acesta este un altgoritm global executat de PKG, care primind ca input parametrul securizat k, va genera o pereche master cheie publică/privată {**msk**, **mpk**} și parametri globali **params** și descrierea spațiului mesajului M_{CLSC}(**params**), spațiul criptotextului C_{CLSC}(**params**) și un spațiul random R_{CLSC}(**params**).

(2) **Extract-Partial-Private-Key**(ID, msk, params): PKG primind de la utilizator identitatea sa ID $\in \{0,1\}^*$ va rula acest algoritm pentru a genera cheia privată parțială, psk_{ID}, pentru identitatea ID.

(3) **Generate-User-Key**(ID, params): acest algoritm rulat de utilizator va genera o valoare secretă x care împreună cu parametri publici a lui PKG, **params**, va genera cheia publică a utilizatorului, pk_{ID}.

(4) **Set-Private-Key**(psk_{ID}, x, params): utilizator va folosi acest algoritm pentru a-și construi cheia privată a sa, sk_{ID}, folosind cheia parțială generată de PKG, psk_{ID}, și valoarea secretă x.

(5) **Signcriptarea**(m, sk_S, ID_S, pk_S, ID_R, pk_R, params, r): după ce transmițătorul va verifica folosind ID_R și parametri publici a lui PKG, **params**, că cheia publică pk_R aparține identității receptorului va signcripta mesajul $m \in M_{CLSC}$(**params**) și va obține criptotextul $\sigma \in C_{CLSC}$(**params**), ce va trimis la receptor.

(6) **Designcriptarea**(σ, sk_R, ID_R, pk_R, ID_S, pk_S, params): receptorul va decripta criptotextul σ folosind cheia lui privată sk_R, și va returna mesajul, m, în cazul în care verificarea chei publice a transmițătorului, sk_S, corespunde identității sale, ID_S, în caz contrar va returna $\perp$.

Un dezavantaj a acestei scheme este datorat faptului că identitatea unui utilizator ID_A nu mai formează cheia publică a sa. Avantajul sistemului constă în faptul

că este posibil să se verifice că cheia publică aparține identității care o identifică, fără a fi necesar un certificat.

În vederea securității ne vom referi la două tipuri diferite de adversari, $\mathcal{A}_{\mathcal{I}}$ și $\mathcal{A}_{\mathcal{II}}$. Adversarul $\mathcal{A}_{\mathcal{I}}$ reprezintă un atacator, o terță parte, împotriva schemei CLSC. $\mathcal{A}_{\mathcal{I}}$ nu are voie să acceseze cheia master, dar poate cere cheia publică și poate să înlocuiască cheile publice cu valorile dorite de el. Adversarul $\mathcal{A}_{\mathcal{II}}$ reprezintă un PKG corupt care generează cheile parțiale ale utilizatorilor. $\mathcal{A}_{\mathcal{II}}$ are acces la cheia master dar nu poate înlocui cheile publice.

Un adversar $\mathcal{A}$ are voie să acceseze următoarele oracole:

(1) **Public-Key-Broadcast-Oracle**: pentru orice identitate ID primită ca input, challengerul va returna cheia publică corespunzătoare. Dacă această cheie nu există challengerul va calcula cheia publică corespunzătoare pk_{ID} pe care o va returna lui $\mathcal{A}$.

(2) **Partial-Private-Key-Oracle**: pentru orice identitate ID primită ca input, challengerul va calcula cheia parțială corespunzătoare psk_{ID} și o va returna lui $\mathcal{A}$.

(3) **Public-Key-Replacement-Oracle**: pentru o identitate ID primită ca input și o cheie publică validă, pk_{ID}, challengerul va înlocui cheia publică curentă cu $pk^*{}_{ID}$.

(4) **Public-Key-Replacement-Oracle**: pentru o identitate ID primită ca input pentru care cheia publică nu a fost înlocuită, challengerul calculează cheia privată sk_{ID} pentru această identitate și o va returna lui $\mathcal{A}$.

(5) **Signcrypt**: primind ca input identitatea transmițătorului, ID_S, identitatea receptorului, ID_R, și un mesaj m, challengerul va rula algoritmul de signcriptare pe mesajul m, folosind cheia privată a transmițătorului, sk_S, și cheia publică a receptorului, pk_R. Este posibil ca challengerul să nu cunoască valoarea secretă a receptorului când cheia publică asociată a fost înlocuită. În acest caz, $\mathcal{A}$ este necesar să ofere cheia privată a receptorului.

(6) **Unsigncrypt**: pentru un criptotext, σ primit ca input, identitatea transmițătorului, ID_S, și identitatea receptorului, ID_R, challengerul va returna rezultatul pentru rularea algoritmului de unsigncrypt pe criptotextul primit, folosind cheia privată a receptorului și cheia publică a transmițătorului.

Este posibil ca challengerul să nu cunoască valoarea secretă a receptorului când cheia publică asociată a fost înlocuită. În acest caz, $\mathcal{A}$ este necesar să ofere cheia privată a receptorului.

Definiția 4.6 Spunem că o schemă de signcriptare certificateless este *IND-CLSC-CCA-sigură*, dacă orice adversar de tip $\mathcal{A}$($\mathcal{A}_{\mathcal{I}}$ si $\mathcal{A}_{\mathcal{II}}$) are un advantaj neglijabil în câștigarea următorului joc.

(1) **Inițializare**: Dacă adversarul este $\mathcal{A}_{\mathcal{I}}$, challegerul va rula algoritmul de *Setup* pentru a genera cheia master secretă msk și cheia master publică mpk, după care transmite mpk lui $\mathcal{A}_{\mathcal{I}}$, și păstrează secretă cheia publică master. Dacă adversarul este $\mathcal{A}_{\mathcal{II}}$, acesta va rula algoritmul de *Setup* pentru a genera cheia master secretă msk și cheia master publică mpk. $\mathcal{A}_{\mathcal{II}}$ va trimite challengerului mpk și msk.

(2) **Faza 1**: În această fază, $\mathcal{A}$ va realiza o serie limitată de cereri la oracole. $\mathcal{A}_{\mathcal{II}}$ nu necesită să acceseze oracul de generare a chei parțiale, acesta poate să o genereze singur.

(3) **Challenge**: La sfârșitul primei faze, adversarul va returna două identități distincte, ID_{S^*} și ID_{R^*} și deasemenea două mesaje de lungime egală { m_0, m_1}. Challengerul va alege random un bit γ și va signcripta m_γ folosind cheia privată a lui ID_{S^*} și cheia publică a lui ID_{R^*} pentru a obține criptotextul σ^*. Challengerul va returna σ^* adversarului.

(4) **Faza 2**: Adversarul continuă să încerce challengerul cu aceleași cereri făcute în Faza 1.

(5) **Răspuns**: Adversarul va returna un bit γ^*. Spunem că adversarul câștigă jocul dacă $\gamma^* = \gamma$ și adversarul îndeplinește următoarele condiții:

- $\mathcal{A}_{\mathcal{I}}$ și $\mathcal{A}_{\mathcal{II}}$ nu pot extrage cheia privată pentru ID_{R^*}.
- $\mathcal{A}_{\mathcal{I}}$ nu poate extrage cheia privată pentru oricare identitate dacă cheia publică corespunzătoare a fost deja înlocuită.
- $\mathcal{A}_{\mathcal{I}}$ nu poate extrage cheia privată parțială a ID_{R^*}, dacă $\mathcal{A}_{\mathcal{I}}$ a înlocuit cheia publică pk_{R^*} înainte de faza challenge.
- În Faza 2, $\mathcal{A}_{\mathcal{I}}$ nu poate face cereri de unsigncrypt pe criptotextul challengelui pentru ID_{S^*} și ID_{R^*} doar dacă cheia publică a transmițătorului sau cea a receptorului folosite să signcripteze m_γ, au fost înlocuite după.

- În Faza 2, $\mathcal{A}_{\mathcal{II}}$ nu poate face cereri de designcriptare pentru criptotextul γ^* pentru ID_{S^*} și ID_{R^*}, și cheia publică pk_{R^*} care au fost folosite pentru signcriptarea mesajului.

Advantajul lui $\mathcal{A}$ este definit $Adv_A = |2Pr|\gamma^* = \gamma| - 1|$.

Definiția 4.7 Spunem că o schemă de signcriptare certificateless este *EUF-CLSC-CMA-sigură*, dacă orice adversar de tip $\mathcal{A}$($\mathcal{A}_{\mathcal{I}}$ și $\mathcal{A}_{\mathcal{II}}$) are un advantaj neglijabil în câștigarea următorului joc.

(1) **Inițializare**: Este aceeași ca la definiția anterioară.

(2) **Cereri**: $\mathcal{A}$ poate să facă o serie de cereri ca la Faza 1, din definiția anterioară. Pentru a avea și securitate interioară, adversarul poate obține acces la cheia privată a receptorului mesajului sincriptat.

(3) **Răspuns**: După o serie de cereri limitate, $\mathcal{A}$ va returna (σ^*, ID_{S^*}, ID_{R^*}), care nu e produs de o cerere la oracolul de signcriptare. Adversarul câștigă dacă rezultatul signcriptării (c^*, ID_{S^*}, pk_{S^*}, sk_{R^*}) nu este simbolul $\perp$ și cererile îndeplinesc următoarele condiții:

- $\mathcal{A}_{\mathcal{I}}$ și $\mathcal{A}_{\mathcal{II}}$ nu pot extrage cheia privată pentru ID_{S^*}.
- $\mathcal{A}_{\mathcal{I}}$ nu poate extrage cheia privată pentru oricare identitate dacă cheia publică corespunzătoare a fost deja înlocuită.
- $\mathcal{A}_{\mathcal{I}}$ nu poate extrage cheia privată parțială a ID_{S^*}.

Probilitatea lui $\mathcal{A}$, de a câștiga jocul anterior este $Succ_A = Pr|\mathcal{A}castiga|$.

Capitolul 5

Variațiile Signcriptării

5.1 Signcriptarea de tip online/offline

Furnizarea unor mecanisme eficiente pentru confidențialitate și autentificare este probabil cea mai importantă cerere în tranzacțile electronice, în special în cazul dispozitivelor mobile sau cartelelor SIM. Din moment ce atacatorii pot accesa ușor nivelul fizic și pot lansa atacuri în astfel de dispozitive, protecția în astfel de cazuri trebuie să fie foarte eficace. Dar, datorită constrângerii asupra resurselor acestor dispozitive, doar operații "ușoare" sunt permise să fie implementate. Pentru acest motiv, eficiența devine principala preocupare în construirea algoritmilor criptografici în astfel de medii. În acest scenariu, *signcriptarea online/offline bazată pe identitate* joacă un rol vital reușind să combine avantajele criptografiei bazate pe identitate cu cele a procesului de *signcriptarea online/offline* în care primitiva online/offline dă voie calculelor costisitoare să fie rezolvate în faza offline iar signcriptarea realizează confidențialitatea și autenticitatea într-un singur pas obținând astfel mai multă eficiență.
Toate aceste proprietăți pot fi realizate folosind *signcriptarea online/offline bazată pe identitate*[16] care este potrivită pentru dispozitive mai slab dotate deoarece în multe aplicații cel care semnează are un timp de răspuns limitat, de îndată ce mesajul este primit.
Semnătura online/offline este formată din două faze. Prima fază este executată offline înainte ca mesajul ce trebuie semnat este primit, și a doua fază este executată online după ce mesajul de semnat este primit.
Ca și în cazul semnăturii online/offlline, o schemă de *signcriptare online/offline*

bazată pe identitate trebuie să fie eficientă în faza online. Astfel toate costurile computaționale costisitoare,de exemplu evaluarea unei exponențieri sau a unei aplicații biliniare, ar trebui rezolvate în faza offline. Deasemenea partea offline nu este responsabilă cu semnarea sau criptarea anumitor mesaje, acestea find disponibile doar în partea online.
O astfel de schemă este alcătuită din următorii cinci algoritmi.

Setare: dat un parametru de securitate, PKG va genera parametri publici și secretul master.

Extragere: dată o identitate ID_S, PKG va returna cheia privată corespunzătoare.

OffSemnare: dată cheia privată a identității ID_S, și identitatea receptorului, ID_R, se va crea semnătura offline $\sigma^{'}$ efectuându-se operațiile costisitoare din semnarea unui mesaj, ca exponiențierile sau evaluările aplicaților biliniare.

OnSigncriptare: dat mesajul m, ID_R receptorului și semnătura offline $\sigma^{'}$ algoritmul va cripta mesajul și va crea semnătura online, σ,din cea offline și va returna criptotextul ce va fi trimis receptorului.

Designcriptare: primind criptotextul, și ID_S transmițătorului, algoritmul va returna mesajul m sau simbolul $\perp$ dacă criptotextul este invalid între ID_S și ID_R.

O aplicație a schemei de *signcriptarea online/offline bazată pe identitate* este securitatea MANET după cum s-a arătat în [17]. MANET este o rețea mobilă ad-hoc, implementată în principal în sistemele de comunicare militare, find o rețea de dispozitive care interacționează dar nu au structură fixă. De obicei este vorba de o serie de dispozitive care comunică wireless. Astfel se pune problema securității în astfel de rețea deoarece este constituită din noduri care nu stau la aceeași distanță de comunicare pentru o perioada lungă. Fiecare nod este liber să se mute în orice direcție, astfel se schimbă legăturile cu celelate dispozitive frecvent. Datorită acestor probleme, performanța acestor rețele este drastic redusă. Problema rutării este rezolvată de folosirea protocolului Ad hoc On-Demand Distance Vector(AODV), prin care fiecare nod trebuie să păstreze informații despre traseul traficului. Chiar dacă rezolvă eficiența, nu se rezolvă și securitatea sistemului. Introducerea unor algoritmi de criptare și semnare complecși vor scădea din nou eficiența protocolului.

Astfel schema propusă în [17] reușeste să rezolve problema eficienței prin folosirea a unei combinații între *signcriptare* și principiul *semnăturii online/offline*. Schema este alcătuită din patru algoritmi: setup, signcriptarea offline, signcriptarea online și verificarea signcriptării. În faza de setare, sistemul generează și publică parameteri publici, și fiecare utilizator își generează propria cheie privată și o cheie de sesiune fără o autoritate de certificare (CA). În faza de signcriptare offline, transmițătorul Alice calculează signcriptarea offline care folosește operațiile cele mai costisitoare din punct de vedere computațional. În faza online, Alice va semna și cripta eficient mesajul, și va trimite apoi textul signcriptat la receptorul Bob. În faza de verificare a signcriptării, receptorul Bob folosește o cheie de sesiune pentru a decripta. Deasemenea va verifica semnătura. Aceste faze se desfășoară astfel:

Setup

Sistemul va genera și va publica parametrii unei curbe eliptice (pentru detalii privind, se poate consulta [6]):

- p este un număr prim reprezentând mărimea câmpului F_p
- $a, b \in F_p$ sunt parametrii curbei eliptice E
- $G \in E$ este un grup de ordin n, cu n număr prim
- h ce măsoară rația dintre ordinul curbei și cel a lui G, $h = (ordE)/(ordG)$
- $H(.)$ o funcție hash one-way
- $E_k()/D_k()$ este un algoritm simetric de criptare/decriptare cu o cheie k

După publicare, nodul care transmite mesajul va obține informații despre toate nodurile vecine pentru a genera o cheie de sesiune ca cea menționată mai jos. Fiecare nod vecin din rețea partajează o parolă secretă. Vom presupune că Alice este nodul ce transmite și Bob, nodul receptor, find unul din nodurile vecine. Ei vor calcula individual două variabile întregi, t și $-t$, find derivate din parola secretă. Astfel pentru a genera cheia de sesiune vor executa următorii pași:
Pasul 1: Alice va genera random un întreg $d_A < n$ ca cheie privată a sa. Va calcula apoi $Q_A = (d_A + t)G$, și apoi Alice va trimite Q_A lui Bob.
Pasul 2: După primire Q_A va genera și el cheia sa privată d_B apoi va calcula
$Q_B = (d_B + t)G$, $Y = Q_A + (-t)G = d_A G$, $K_B = d_B Y = d_A d_B G$;
$tK_B = td_A d_B G$ și va trimite Q_B, tK_B lui Alice.
Pasul 3: Alice primește Q_A si tK_B, și va calcula apoi $X = Q_B + (-t)G = d_B G$,
$K_A = d_A d_B G$; și dacă $tK_A = tk_b$, va trimite $td_B G$ lui Bob.
Pasul 4: Bob va verifica $td_b G$.

După ce procedura de verificare a fost finalizată de amândoi, aceștia sunt gata să înceapă să folosească cheia de sesiune.

Signcriptarea offline

Pasul 1: se va genera aleator un întreg $r < n$

Pasul 2: va returna signcriptarea offline (R, K), unde $R = rG = (r_1, r_2)$, K cheia de sesiune.

Signcriptarea online

Pasul 1: Dat un mesaj M, și signcriptarea offline (R, K), va cripta mesajul folosind algoritmul simetric de criptare generând criptotextul $C = E_K(M)$.

Pasul 2: va folosi funcția hash și va calcula $h = H(M||r_1)$

Pasul 3: va calcula $S = (r + hd_A r_1) \bmod n$

Astfel va rezulta tripleta (C, R, S) ca rezultat a signcriptării, aceasta find trimisă lui Bob

Verificarea

Dupa ce primește tripleta (C, R, S) va recupera criptotextul C executând algoritmul simetric de decriptare cu cheia de sesiune și va verifica deasemenea semnătura, după următorii pași:

Pasul 1: va recupera mesajul M, folosind cheia de sesiune $M = Dec_K(C)$

Pasul 2: va folosi funcția hash one-way pentru a calcula $h = H(M||r_1)$

Pasul 3: va verifica dacă $R = SG$ - hr_1Y. În caz afirmativ va accepta M ca mesaj trimis de Alice, în caz contrar va respinge M.

Din punct de vedere a eficienței, cele mai costisitoare operații din punct de vedere computațional sunt înmulțirile. În total sunt 7 astfel de operații, majoritatea în algoritmul de setup. Însă aceasta schemă se evidențiază de celelalte scheme de acest fel, prin faptul că costul online poate fi neglijat, nici o operație costisitoare nu are loc în acest algoritm.

Din punct de vedere a securității schema asigură confidențialitatea chiar dacă cheia secretă este aflată, deoarece pentru ca să o poată calcula are nevoie să cunoască t care este cunoscut doar de Alice și Bob. Deasemenea se asigură integritatea datelor și autentificarea prin parola secretă.

5.2 Signcriptarea partajată

Schemele și aplicațiile prezentate până acum implică entități individuale (un singur receptor, un singur emițător). În multe cazuri, trebuie interzis accesul doar unui receptor la mesajul signcriptat. O astfel de schemă a fost prima dată propusă de Shamir în [9], care a dezvoltat o metodă simplă și elegantă de partajare a unor informații secrete între n membri, astfel încât pentru recuperarea informațiilor secrete este necesară cooperarea a cel puțin t dintre ei.
Prima schemă de signcriptare folosind principiul secretului partajat a fost propusă de Zhang în [10] ca o alternativă la schemele anterioare, care prezentau o serie de probleme ca prevenirea unor receptori să înșele pe ceilalți sau eficiența, folosindu-se încă metoda clasică, "semnează și apoi criptează".
Chiar dacă a rezolvat problemele anterioare, și aceasta se confruntă cu o serie de probleme ca *managementul cheilor* sau *verificarea publică*, care nu este posibilă deoarece doar receptorii pot verifica semnătura, deoarece la operația de Designcriptare este necesară cheia privată a sa, astfel nu se poate asigura $non - repudierea$.
În [18] este prezentată o schemă de signcriptare partajată bazată pe identitate cu designcriptare partajată. În această schemă un grup de u membri sau mai mulți pot coopera pentru ca mesajul să fie signcriptat, dar $u-1$ membri sau mai puțini nu pot reuși acest lucru. Pe partea de designcriptare orice entitate externă poate verifica valabilitatea semnăturii, dar pentru a recupera mesajul cooperarea a cel puțin t membri a grupului receptor este necesară. Această schemă rezolvă și problemele schemei anterioare asigurând verificabilitatea publică și simplificând managementul cheilor prin folosirea criptografiei bazate pe identitate.
Schema propusă se realizează între trei entități, PKG, grupul $\mathcal{A}$ format din m transmițători A_1, $A_2...A_m$, și grupul receptor $\mathcal{L}$, format din n membri L_1, $L_2...L_n$. Se alege u(prag) și m astfel încât $1 \leq u \leq m \leq q$ și deasemenea t(prag) și n să satisfacă $1 \leq u \leq m \leq q$. Astfel un mesaj este signcriptat împreună de cel puțin u membri ai grupului $\mathcal{A}$ și designcriptat împreună de cel puțin t membri ai grupului $\mathcal{L}$.
Această schemă este formată din cinci algoritmi:

Setup: dat un parametru de securitate k, PKG va genera parametri publici a sistemului. Deasemenea va calcula cheia master s ce este ținută secretă și va produce o cheie P_{pub}, care este cheia publică a grupului $\mathcal{A}$.

Extragere: dată o identitate ID, și o cheie privată S_{ID}, m și pragul u, acest algoritm generează m părți a chei private S_{ID} și furnizează câte una fiecărui membru din grupul $\mathcal{A}$. Deasemenea generează un set de chei de verificare care pot fi folosite pentru a verifica fiecare parte din cheia privată. Astfel fiecare membru va primi o parte din cheia privată care o va ține secretă și cheia de verificare corespunzătoare care o va afișa public.

Signcriptare: în acest algoritm se folosește un manager de grup de încredere pentru a signcripta mesajul. Astfel, pentru ca un mesaj să fie signcriptat se va face o cerere la menagerul de încredere. Acesta va trimite această cerere la t membri a grupului $\mathcal{A}$ care vor signcripta individual mesajul. După ce va primi toate signcriptările individuale, managerul va verifica valabilitatea acestora. Dacă toate sunt valide, va crea o signcriptare finală combinând toate signcriptările individuale. Acesta va trimite criptotextul membrilor grupului receptor $\mathcal{L}$.

Designcriptare: pentru a recupera mesajul din criptotextul primit vor colabora cel puțin t membri a grupului $\mathcal{L}$.

O astfel de schemă reușeste să atingă următoarele proprietăți de securitate:
• *Proprietatea de unforgeability* este asigurată prin faptul că un grup ce conține un număr maxim de $u-1$ membri nu vor putea să falsifice un text signcriptat valid.
• *Confidențialitatea* este asigurată prin faptul că un grup format din $t-1$ membri nu vor putea să recupere mesajul.
• *Verificabilitatea publică universală* se asigură prin faptul că verificarea semnăturii se poate face de orice entitate care are parametri publici a sistemului.

O astfel de schemă este ideală în aplicații în care participanți pot coopera pentru a obține un avantaj peste ceilalți. Folosind schema propusă se poate contracara acest fenomen. De exemplu, se poate folosi în licitații electronice pentru a preveni colaborarea casei de licitație cu o parte din licitanți.

5.3 Signcriptarea de tip inel

Să ne imaginăm un scenariu în care un membru a unui cabinet vrea să ofere o informație importantă presei, privind președintele țării. El trebuie să ofere informații secrete, într-o manieră anonimă, altfel ar putea să se expună pericolului. Presa nu va accepta informația decât dacă este autentificată de unul dintre membri cabinetului. Dacă informația este foarte importantă și nu trebuie scăpată publicului până când va ajunge în mâna presei, ar trebui ca aceasta să fie transmisă în mod confidențial. Astfel este nevoie de *anonimitate* pentru protecția membrului cabinetului care transmite informația, $autentificare$ pentru ca presa să înteleagă că informația provine de la o sursă sigură, și *confidențialitate* până când informația va ajunge în mâinile presei. Toate aceste informații sunt realizate de o singură primitivă criptografică numită *Signcriptarea de tip inel*.
Signcriptarea de tip inel are la bază semnătura inel, care este o variație la semnătura grup. Semnătura grup este caracterizată printr-un manager de încredere care definește grupuri de utilizatori și distribuie chei speciale membrilor grupurilor. Membrii individuali a grupurilor pot să folosească cheile primite și să semneze anonim mesaje din partea grupului din care fac parte. Semnătura produsă de membri grupurilor este imposibil de identificat de către cei care o verifică, dar nu și de către managerul grupurilor, care poate revoca anonimitatea membrilor care nu respectă regulile. Astfel semnătura inel a apărut ca răspuns la puterea pe care o are managerul, aceasta implicând doar utilizatorii. Deasemenea nu necesită o coordonare între utilizatori, aceștia nu necesită să se cunoască între ei. Semnăturile se pot crea "ad-hoc", utilizatori ne mai fiind restrânși la anumite grupuri. Astfel această semnătură este utilă când utilizatori nu vor să colaboreze, iar semnătura grup, pentru atunci când aceștia doresc să colaboreze.
Signcriptarea de tip inel folosește aceleași principii ca și semnatura inel, aceasta find o schemă de signcriptare anonimă care dă voie unui utilizator să signcripteze un mesaj anonim din partea unor utilizatori(membri a unui inel). Astfel cine va verifica mesajul este asigurat că mesajul signcriptat provine de la unul din membri inelului, dar nu dezvăluie nici o informație despre care membru este vorba.
În [19] este prezentată o schemă de *signcriptare de tip inel* bazată pe identitate care este formată din următorii algoritmi:

Setup: dat un parametru de securitate k, PKG generează parametri publici ai sistemului. Deasemenea produce și cheia master care e păstrată secretă.

Extragere: dată o identitate ID, PKG va calcula cheia privată corespunzătoare D_{ID} și o va transmite în mod securizat proprietarului.

Signcriptare: pentru a transmite un mesaj m receptorului Bob a cărui identitate este ID_B, Alice va alege un grup $U = \{u_1, u_2, ...u_n\}$ cu identitățile $\{ID_1, ID_2, ...ID_n\}$ printre care este și ea. Aceasta va rula algoritmul $Signcriptare(m, D_{ID_A}, U, ID_B)$ din partea grupului în care va folosi cheile publice a grupului U și va obține criptotextul σ.

Designcriptare: după ce Bob va primi criptotextul σ acesta va rula algoritmul $Designcriptare(\sigma, U, D_{ID_B})$ și va obține mesajul m sau simbolul $\perp$ dacă criptotextul σ este invalid relativ la grupul U și Bob.

Autentificare: acest algoritm poate fi folosit de către Alice, pentru a demonstra că ea este utilizatorul care a transmis criptotextul σ. Algoritmul este restricționat doar la utilizatorul care transmite criptotextul deoarece necesită cheia privată a utilizatorului.

Signcriptarea de tip inel poate fi folosită în diverse aplicații cum ar fi licitațiile electronice. Licitațiile electronice sunt împărțite în două: publice, în care toate prețurile sunt la vederea participanților aceștia find nevoiți în runde multiple să crească prețul pentru a câștiga și de tip închise, în care participanți licitează toți o singură dată. Ofertele lor sunt individuale, acestora fiindu-le interzis să coopereze între ei. Într-o astfel de licitatie, oferta cea mai mare câștigă. Însă, pentru ca o astfel de licitație să fie securizată sunt necesare următorarele cerințe:(1)să fie corectă, (2) să se pastreze secret ofertele, (3) să se asigure non-repudierea, (4) anonimitatea, (5) verificabilitatea, (6)confidențialitatea participanților, (7) proprietatea de unforgeability. Toate aceste condiții sunt îndeplinite în schema [20], în care este propusă și eliminarea casei de licitație, datorită problemelor în schemele anterioare în care casa de licitație putea colabora cu participanți.
O astfel de schemă constă dintr-un cumpărător, care are nevoie de cineva pentru efectuarea unei serviciu, și o mulțime de vânzători, *licitanți*, care își oferă serviciul. Va câștiga cel cu oferta cea mai mică.
Schema se desfășoară astfel:

Faza inițială în care cumpărătorul, va publica informațiile necesare și condițiile de îndeplinit. O parte din vânzători vor răspunde la acest anunț. Apoi cumpărătorul va alege pe cei care corespund cel mai bine criteriilor sale.

Faza licitației în care fiecare vânzător ales va face o ofertă, care va fi signcriptată și va fi trimisă la cumpărător.

Faza de deschidere în care după ce cumpărătorul a primit toate ofertele, va recupera valoarea fiecărei oferte și va notifica valoarea ofertei minime împreună cu oferta signcriptată corespunzătoare tuturor licitanților, pentru ca câștigătorul să confirme oferta.

Faza de confirmare a câștigătorului în care licitanții află valoarea ofertei câștigătoare printr-o notificare. Cel care știe că a câștigat va trimite un mesaj de confirmare la cumpărător cu datele necesare pentru ca cumpărătorul să-l poată identifica ca câștigător. Dacă datele primite îl identifică ca cel care a trimis oferta minimă, acesta este declarat câștigător.

Faza de confirmare a pierzătorilor în care dacă câștigătorul nu a confirmat încă oferta, pierzătorii vor trebui să confirme fiecare că nu este câștigător. Adevăratul câștigător nu poate confirma acest lucru și după ce toți ceilalți au confirmat acest lucru, câștigătorul este găsit.

Acest protocol asigură toate cerințele de securitate cerute de o licitație de tip închisă :
• *corectitudinea* prin eliminarea casei de licitație care putea colabora cu participanți la licitație
• *păstrarea secretă a ofertelor* prin folosirea unei funcții hash one-way
• *asigurarea non-repudierei* prin faptul că participanți pot confirma că nu sunt câștigători
• *anonimitatea* datorită folosirii unei scheme de signcriptare de tip inel
• *verificabilitatea* prin publicarea prețului câștigător și a ofertei signcriptate
• *confidențialitatea participanților* datorată proprietății semnăturii inel
• *proprietatea de unforgeability*, falsificarea unei semnături este intractabilă

Schema propusă în [20] are următorii parametri:

i $\in \{1,2, \ldots n\}$	vânzătorii
A	cumpărătorul
$\lambda \in \mathbb{N}$	parametru de securitate
$\mathbb{G}_1$, $\mathbb{G}_2$	două grupuri multiplicative
q	număr prim mare
P	generator de $\mathbb{G}_1$
e: $\mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$	aplicație bilineară
H, H_1, H_2, H_3	$H_i : \{0,1\}^* \to \mathbb{Z}_q (i = 1,2,3)$
$x_i \in \mathbb{Z}_q$	cheia privată a vânzătorului i
$y_i = x_i P$	cheia publică a vânzătorului i
$x_A \in \mathbb{Z}_q$	cheia privată a cumpărătorului A
$y_A = x_A P$	cheia publică a cumpărătorului A
V_i	prețul vânzătorului i
C_i	prețul criptat a vânzătorului i
BB	bulletin board

Faza inițială

Cumpărătorul(A) va publica informațiile relevante către toți vânzătorii, de exemplu, detalii despre un proiect, precum cerințele și condițiile corespunzătoare. După, o parte din vânzători vor răspunde lui A, iar acesta va selecta pe cei mai potriviți vânzători care satisfac cerințele și condițiile. Astfel vor rezulta n vânzători care vor participa la licitație, $i \in \{1,2...n\}$

Faza de licitatie

- **Pasul 1 :** vânzătorul k va genera $d \in_R \mathbb{Z}_q$, va calcula $g = e(P,P)$, $\alpha = g^d$, $C_k = H_1(\alpha) \otimes V_k \in \{0,1\}^\lambda$, $\beta = d y_A$ și $c_k = H(V_k)$;

- **Pasul 2 :** vânzătorul k va genera $r_0 \in \{0,1\}^\lambda$, $L = y_1 \| y_2 \| ... \| y_n$, va calcula $\mu_0 = H_2(0, r_0, V_k, L)$, $\mu_1 = H_2(1, r_0, V_k, L)$, $\rho_k = e(\mu_0, \mu_1)^{x_k}$;

- **Pasul 3 :** vânzătorul k va genera $t, r_1 \in_R \mathbb{Z}_q$, va calcula $M = e(P,P)^t$, $N = e(\mu_0, \mu_1)^t$, $R = {\rho_k}^{r_1}$;

- **Pasul 4 :** vânzătorul k va lua $U_i \in \mathbb{G}_1$, $i \neq k$, va calcula $h_i = H_2(V_k, M, N, R, \rho_k, U_i)$, $U_k = r_1 y_k$ - $\sum_{i \neq k} (U_i + h_i y_i - h_i y_k)$,

$h_k = H_2(V_k, M, N, R, \rho_k, U_k)$, $s = t$ - $(r_1 + \sum_{i=1}^{n} h_i)x_k$;

- **Pasul 5 :** vânzătorul k trimite signcriptarea $\sigma_k = (C_k, c_k, \beta, \rho_k, r_0, \pi_1)$, unde $\pi_1 = (M, N, R, \{U_i\}_{i=1}^{n}, s)$ lui A ;

Faza de deschidere

După ce A primește toate signcriptările σ_i , $i \in \{1,2 \ ... \ n\}$, A va urma pași următori:

- **Pasul 1 :** va calcula $\alpha^{'} = e(\beta, \frac{1}{x_A}P)$, $V_i^{'} = H_1(\alpha^{'}) \otimes C_i \in \{0,1\}^{\lambda}$, $i \in \{1,2 \ ... \ n\}$, $c_i^{'} = H(V_i^{'})$ și va alege prețul minim $V_i^{'}$ ca ofertă câștigătoare dacă $c_i^{'} = c_i$.

- **Pasul 2 :** va calcula $h_i^{'} = H_2(V_i^{'}, M, N, R, \rho_i, U_i)$, $i \in \{1,2 \ ... \ n\}$, $\mu_0 = H_2(0, r_0, V_i^{'}, L)$, $\mu_1 = H_2(1, r_0, V_i^{'}, L)$ și va verifica

  (1) $M = e(P,P)^s \cdot e(P, \sum_{i=1}^{n}(U_i + h_i^{'}y_i))$

  (2) $N = \rho_i^{\sum_{i=1}^{n} h_i^{'}} \cdot R \cdot e(\mu_0, \mu_1)^s$

- **Pasul 3 :** Dacă condițiile (1) și (2) sunt adevărate, A va accepta signcriptarea σ_i și va publica oferta câștigătoare și signcriptarea acesteia, $(V_i^{'}, \sigma_i)$; astfel A va alege următorul preț minim $V_i^{'}$ până când ambele condiții sunt adevărate. În cele din urmă dacă câștigătorul este vânzătorul j, atunci prețul câștigător și signcriptarea corespunzătoare va fi $(V_j^{'}, \sigma_j)$.

Faza de confirmare

Vânzătorii cunosc oferta câștigătoare și signcriptarea corespunzătoare prin BB. Câștigătorul j va folosi protocolul de confirmare să demonstreze faptul că σ_i este produsă de el :

- **Pasul 1 :** va genera $t^{'} \in_R \mathbb{Z}_q$ și va calcula $M^{'} = e(P,P)^{t^{'}}$, $N^{'} = e(\mu_0, \mu_1)^{t^{'}}$, $h_j^{'} = H_2(M^{'}, N^{'}, \rho_j)$, $s^{'} = t^{'}$ - $h_j^{'}x_j$ și va trimite $\pi_2 = (M^{'}, N^{'}, s^{'})$ la BB.

- **Pasul 2 :** A și verificatori vor calcula $h_j^{'} = H_2(M^{'}, N^{'}, \rho)$ și vor verifica următoarele ecuații

  (1) $M^{'} = e(P,P)^{s^{'}} \cdot e(P, y_j)^{h_j^{'}}$

  (2) $N = \rho^{h_j^{'}} \cdot e(\mu_0, \mu_1)^{s^{'}}$

  Dacă condițiile (1) și (2) sunt adevărate atunci verificatori și A vor accepta vânzătorul j ca câștigător.

Faza de confirmare a pierzătorilor

Când nimeni nu confirmă oferta, vânzătorii trebuie să confirme că ei nu sunt căștigătorul, însă acest lucru nu poate fi confirmat de câștigător. După ce $n-1$ vânzători au executat protocolul de confirmare a pierzătorilor, câștigătorul va fi gasit. De notat că nu este nevoie de o terță parte pentru arbitrare. Protocolul este următorul:

Pentru ca un vânzător $l \in L$ să confirme că nu el a generat $(V_j^{'}, \sigma_j)$ va urma pași următori:

- **Pasul 1 :** va calcula $\rho_l = e(\mu_0, \mu_1)^{x_l}$ și va generara $t^{'} \in_R \mathbb{Z}_q$ și va calcula $M^{'} = e(P,P)^{t^{'}}$, $N^{'} = e(\mu_0, \mu_1)^{t^{'}}$, $h_l^{'} = H_2(M^{'}, N^{'}, \rho_l)$, $s^{'} = t^{'}$ - $h_l^{'} x_l$ și va trimite $\pi_3 = (M^{'}, N^{'}, s^{'})$ la BB.

- **Pasul 2 :** A și verificatori vor verifica legătura dintre ρ_l și ρ_j , și va verifica

  (1) $M^{'} = e(P,P)^{s^{'}} \cdot e(P, y_l)^{h_l^{'}}$

  (2) $N = \rho^{h_l^{'}} \cdot e(\mu_0, \mu_1)^{s^{'}}$

  Dacă condițiile (1) și (2) sunt adevărate și $\rho_l \neq \rho_j$, atunci A și verificatori acceptă faptul că vânzătorul l nu este câștigătorul.

Următoarele cerințe de securitate sunt asigurate de această schemă de licitație :

• *corectitudinea* în schema propusă se asigură prin faptul că nimeni nu poate afla informații suplimentare despre procesul licitației. Deasemenea nu este nevoie de o terță parte în protocol, și A nu poate conspira cu vânzători, deoarece nu este în interesul lui.

• *păstrarea secretă a ofertelor* se face prin folosirea unei funcții hash one-way, astfel prețul signcriptării $C_k = H_1(\alpha) \otimes V_k \in \{0,1\}^{\lambda}$ este păstrat secret.

• *asigurarea non-repudierei* prin faptul că schema este bazată pe condiția de anonimitate a signcriptării inel, astfel când câștigătorul nu vrea să confirme oferta,

ceilalți participanți pot confirma că nu sunt câștigătorii prin faza de confirmare a pierzătorilor, astfel se va afla câștigătorul.

• *anonimitatea* se asigură în faza de licitație, unde se adoptă o semnătură inel, astfel încât identitatea vânzătorilor este protejată de semnătura inel.

• *proprietatea de unforgeability*, se asigură prin faptul că adversarul nu poate falsifica cheia privată.

• *confidențialitatea participanților* în această schemă se asigură prin faptul că nimeni nu poate afla informații despre vânzător dintr-o ofertă, cu excepția valorii ofertei. Chiar dacă cumpărătorul poate decripta fiecare ofertă, acesta nu poate lega oferta cu o identitate. Astfel, schema asigură confidențialitatea participanților.

• *verificabilitatea* se asigură prin publicarea prețului câștigător și a ofertei signcriptate corespunzătoare.

5.4 Signcriptarea proxy

Signcriptarea proxy a apărut ca măsură pentru transmiterea securizată și autentificată a mesajelor[21] de către calculatoarele din rețea cu capacitate computațională mică. Multe dintre dispozitivele folosite, de exemplu telefoanele mobile intră în această categorie. Puterea computațională redusă a acestor dispozitive, face dificilă efectuarea calculelor matematice grele necesare de unele primitive criptografice ca semnăturile digitale. S-a luat în considerare un model generic de comunicare pentru dispozitivele cu putere computațională mică, în care unul dintre dispozitive poate transmite și primi mesaje în mod securizat de la o serie de alte calculatoare. S-a identificat faptul că semnarea digitală a unui mesaj este operația cea mai costisitoare din punct de vedere computațional în transmiterea mesajelor. Astfel s-a hotărât folosirea a unei scheme de semnare proxy, putându-se încărca și rezolva, calculele computaționale grele de pe dispozitivele cu putere computațională mică pe servere mai puternice. Astfel combinând această primitivă de semnare cu o primitivă de criptare s-a creat prima schemă de *signcriptare proxy*.

Într-o schemă de *signcriptare proxy*, expeditorul original Alice, trimite un mesaj cu semnătura sa către un proxy, Bob, delegând astfel capacitățile sale. Acesta din urmă va folosi informațiile primite și va construi o cheie proxy privată. Bob va putea să efectueze operația de signcriptare pe unele mesaje, care vor apărea

ca find din partea lui Alice. O astfel de schemă este utilă în aplicațiile în care mesajele sunt semnate individual și nu sunt legate de o cheie de sesiune pentru a furniza autenticitatea și integritatea.
O schemă de *signcriptare proxy* este formată din patru algoritmi:

Setup: în care Alice crează secretul proxy cel va trimite agentului de încredere proxy. Acesta îl va folosi pentru a construi cheia secretă proxy.

Signcriptare: pentru ca Alice să trimită un mesaj m lui Bob, îl va cripta și-l va trimite agentului proxy printr-un canal securizat. Acesta va recupera mesajul și va verifica integritatea lui. Dacă totul este în regulă, acesta va efectua operația de signcriptare pe mesajul m folosind cheia privată proxy și obținând criptotextul signcriptat care îl va trimite lui Bob, ca fiind din partea lui Alice.

Designcriptare: Bob va recupera mesajul din criptotextul primit folosind cheia publică a lui Alice și cheia lui privată și va accepta mesajul doar dacă se verifică faptul că semnătura pe mesaj aparține lui Alice.

O astfel de schemă de *signcriptare proxy* este extrem de potrivită pentru protejarea agenților mobili [22]. Aceștia sunt entități care sunt capabile să se mute de pe un calculator pe altul continuând execuția pe calculatorul destinație. Mai bine spus un agent mobil este un proces care își transportă starea dintr-un mediu în altul, păstrând datele intacte, și find capabil să ruleze în noul mediu. Aceștia sunt definiți ca obiecte și au compartament, stare și locație. Se pot executa oricând sau își pot suspenda execuția, și să se mute prin internet la alt host unde își vor continua execuția. Când un agent mobil decide să se mute, își salvează starea curentă, transportă starea salvată la noul host, și va relua execuția din starea salvată.
Agenți mobili reprezintă una dintre cele mai bune aplicații pentru schema semnăturii proxy, deoarece semnătorul original(clientul) trebuie să-și delegheze capacitățile de semnare agentului mobil pentru a putea executa operații de autentificare din partea clientului. Însă, semnătura proxy oferă doar autenticitate și integritate mesajului. Poate exista și nevoia confidențialității mesajului dacă agentul mobil transportă mesaje importante. O soluție pentru a satisface proprietățile unui mesaj este metoda "semnează și apoi criptează". După cum am văzut, această metodă nu este foarte eficientă din punct de vedere computațional. Astfel, folosirea schemei de *signcriptare proxy* este esențială în contextul agenților mobili. Agenți

mobili pot fi folosiți într-o arie largă de aplicații, cum ar fi căutarea și filtrarea de informații, multimedia, în aplicații militare, telecomunicație. Abilitatea agentului mobil de a acționa și negocia din partea creatorilor îi fac extremi de potriviți pentru comerțul electronic. De exemplu, în loc să petreci mult timp pentru căutarea celei mai bune oferte la o carte într-un magazin online, construind un astfel de agent vei salva un timp considerabil. Agentul ar putea fi programat să viziteze mai multe magazine și să găsească cea mai bună ofertă pentru cartea pe care o dorești.
Un astfel de protocol de căutare implică o autoritate de sistem (SA), un utilizator (U), un host(H) și un agent mobil (MA) generat de utilizator în care se folosesc următorii parametri:
$E(F_{3^m})$: este o curbă eliptică unde E: $y^2 = x^3$ - x + 1(mod 3^m)
$\mathbb{G}_1$ un grup aditiv de ordin q
P un generator al lui $\mathbb{G}_1$ a cărui ordin este q
$\mathbb{G}_2$ un grup multiplicativ de E a cărui ordin este q
$\hat{e}$ o aplicație bilineară $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$
M conținutul unui contract a cărui mărime este n
m_w cererile lui U(include descrierea produselor, prețul maxim, identitatea utilizatorului ID_U, cheia publică a utilizatorului Y_U și alte informații publice)
H_1 o funcție hash one-way $H_1 : \{0,1\}^* \to \mathbb{G}_1$
H_2 o funcție hash one-way $H_2 : \{0,1\}^* \to {\mathbb{Z}_q}^*$
H_3 o funcție hash one-way $H_3 : \mathbb{G}_2 \times \mathbb{G}_1 \to \{0,1\}^{2l+n}$, unde l este numărul de biți necesari de a reprezenta un element a lui $\mathbb{G}_1$

Astfel acest protocol este format din următoarele faze:

Faza de setare în care SA generează cheia master $s \in {\mathbb{Z}_q}^*$ și calculează cheia publică a sistemului $P_{pub} = sP$. Apoi SA va publica parametri publici ai sistemului $\{E, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, P, P_{pub}, H_1, H_2, H_3\}$ și va ține secret cheia master.

Faza de înregistrare în care utilizatorul va executa pașii următori pentru a se înregistra la SA, și pentru a obține cheia publică și privată corespunzătoare.
1. U alege o cheie privată $x_U \in {\mathbb{Z}_q}^*$ și va calcula $Y_U = x_U P$. Apoi U va trimite (ID_U, Y_U) la SA.
2. SA va calcula $Q_U = H_1(ID_U, Y_U)$ și va calcula $S_U = sQ_U$ ce va fi trimis în mod securizat lui U

3. U poate valida S_U verificând condiția $\hat{e}(S_U, P, P) = \hat{e}(Q_U, P_{pub})$. Dacă condiția este adevărată atunci U acceptă (x_U, S_U) ca cheie privată și Y_U ca cheie publică. Similar, hostul H se va înregistra la SA și va obține cheia sa privată (x_H, S_H) și cheia publică Y_H corespunzătoare.

Faza de creare a agentului mobil în care U va genera o semnătură pe m_w:
1. Va genera aleator r din $\mathbb{Z}_q{}^*$ și va calcula $T = rQ_U$.
2. Va calcula $W = x_U H_1(m_w, T)$, $h = H_2(m_w, T, W)$ și $V = (r + h)S_U$
U va trimite lista de cereri (m_w, T, W, V) agentului mobil și-l va lansa în execuție pentru căutarea unui anumit produs pe internet.

Faza de execuție a agentului mobil în care agentul mobil găsește produsul ce corespunde specificaților lui U, va încerca să negocieze signcriptarea unui contract M cu hostul care deține produsul. Pentru acest lucru se realizează următorii pași:
1. H va calcula $h = H_2(m_w, T, W)$ și va verifica dacă $\hat{e}(P, V) = \hat{e}(P_{pub}, T + hQ_U)$ si $\hat{e}(P, W) = \hat{e}(Y_U, H_1(m_w, T))$ pentru validarea semnăturii
2. Dacă semnătura este validă, H va calcula $S_P = V + hS_H$ ca cheie proxy privată și $Q_P = T + h(Q_U + Q_H)$ ca cheie proxy publică, altfel H se va opri.
3. H va genera signcriptarea pe contractul M urmând pași următori:
- Va genera aleator t din $\mathbb{Z}_q{}^*$
- Va calcula $X_1 = tQ_P$ si $X_2 = Y_H$
- $Y = x_H H_1(M, X_1, X_2, Y)$
- $v = H_2(M, X_1, X_2, Y)$
- $Z = (t + v)S_P$
- $w_1 = \hat{e}(tS_P, Q_U)$ și $w_2 = tx_H Y_U$
- $y = H_3(w_1||w_2) \otimes (Y||Z||M)$
4. H va atașa mesajul signcriptat (X_1, X_2, y, m_w, T, W) agentului mobil și-l va trimite înapoi la U.

Faza întoarcerii agentului mobil în care după ce U primește mesajul criptat de la host, acesta îl va decripta și va verifica semnătura pe mesaj
1. U va calcula $w_1 = \hat{e}(X_1, S_U)$ și $w_2 = (x_U, X_2)$
2. Va recupera $Y||Z||M = y \otimes H_3(w_1||w_2)$
3. Va calcula $Q_P = T + h(Q_U + Q_H)$, unde $h = H_2(m_w, T, W)$
4. Va calcula $v = H_2(M, X_1, X_2, Y)$ și va verifica dacă $\hat{e}(P, Z) = \hat{e}(P_pub, X_1 + vQ_P)$

și $\hat{e}(P, Y) = \hat{e}(Y_H, H_1(M, X_1, X_2))$. Dacă ambele condiții sunt adevărate U va accepta mesajul de la hostul H.

5.5 Signcriptarea oarbă

Sistemele de plăți electronice sunt unele din cele mai importante aplicații de pe Internet. În astfel de sisteme utilizatori pot retrage bani de la o bancă în mod electronic, și apoi pot plăti comercianți folosind bani retrași preferabil fără a comunica cu banca în timpul plății. În final, comerciantul va depune bani primiți în bancă. În astfel de sisteme, un aspect important este păstrarea intimități clientului asupra celor cumpărate de pe Internet, pentru aceasta trebuie asigurată anonimitatea identității în toate plățile sale pentru a nu se putea lega identitatea clientul de vreun produs sau serviciu cumpărat. Pentru a se putea asigura acest lucru un nou tip de semnătură s-a creat, semnătura oarbă.
O astfel de semnătură asigură anonimitatea utilizatorilor asigurând posibilitatea unui utilizator să obțină o semnătură validă pentru un mesaj de la un semnător, fără ca cel care semnează să vadă conținutul mesajului. Astfel spus, să presupunem că Alice are o scrisoare care trebuie semnată de o autoritate de încredere, Bob, dar Alice nu vrea ca Bob să vadă mesajul scrisorii. Pentru aceasta ea pune scrisoarea într-un plic care conține și o hârtie indigo și îl trimite lui Bob. Bob va semna pe plic fără să-l deschidă și-l va trimite înapoi. Alice poate apoi deschide plicul și va găsi scrisoarea semnată de Bob, fără ca acesta să fi văzut conținutul[27].
Astfel folosind acest principiu schemele de *signcriptare oarbă* se integrează perfect în aplicații de plată electronică[23], care pe lângă anonimitatea utilizatorilor reușeste să asigure confidențialitate și integritate conținutului mesajului care sunt extrem de importante în astfel de aplicații.
O astfel de schemă implică trei entități: cel care semnează, cel care are nevoie de semnătură și cel care primește mesajul. Într-o aplicație de plată electronică, participanții sunt banca, clientul și comerciantul. Clientul este cel care are nevoie de o semnătură oarbă de la bancă pentru efectuarea unei plăți către un comerciant. Pentru o astfel de aplicație se poate observa că semnătura băncii trebuie să poată fi verificată public. Este necesar ca mesajul cu plata să nu poată fi citit de altcineva decât de comerciantul căruia îi este adresat, fiind necesară asigurarea confidențialități mesajului. Deasemenea comerciantul ar trebui să poată verifica faptul că mesajul nu a fost modificat în timpul transmisei rezultând faptul că și

proprietatea de integritate este necesară.

După cum am spus o astfel de schemă implică trei părți A(banca), B(clientul) și C(comerciantul). Schema este formată din următorii algoritmi:

Setup în care pentru un parametru de securitate k, algoritmul va genera parametri publici a sistemului, printre care două numere prime mari p și q, cu q care se împarte la p-1 și $q > 2^k$, un generator g de subgrupuri $\mathbb{Z}_p$ de ordin q, o funcție hash one-way cu cheie, $KH : \{0,1\}^* \times \mathbb{Z}_q^* \times {\mathbb{Z}_q}^* \rightarrow {\mathbb{Z}_q}^*$, și un sistem de criptare cu cheie privată.

Generare Chei în care fiecare utilizator care dorește să se alăture sistemului generează aleator un întreg $x \in \mathbb{Z}_q^*$ și calculează cheia publică corespunzătoare $y = g^x$ pentru care va trebui să obțină un certificat de autoritate pentru ea de la o autoritate de certificare(CA).

Signcriptarea oarbă se realizează interactiv între entitățile A și B, care rezultă într-un criptotext și o semnătură pentru un mesaj m de la entitatea A să fie transmis la entitatea C:
1. Entitatea B va trimite o cerere de semnătură lui A
2. Entitatea A va genera un întreg $\hat{x} \in \mathbb{Z}_q^*$ și va calcula $\hat{t} = g^{\hat{x}} \ mod \ p$ să fie trimis entității B. Între timp, entitatea B va alege două elemente δ și γ aparținând lui $\mathbb{Z}_q^*$ și va calcula $t^* = g^{\delta} {y_a}^{\gamma} \ mod \ p$.
3. După ce primește $\hat{t}$, B va calcula $t = \hat{t}t^* \bmod p$
4. Apoi B va alege un număr întreg $u \in \mathbb{Z}_q^*$ și va calcula $k_1 = g^u \ mod \ p$ și $k_2 = {y_c}^u \ mod \ p$
5. B va calcula $r = KH(m, k_1, t)$ și $\hat{r} = (r \text{ - } \gamma) \bmod q$, unde m este mesajul. Valoarea lui $\hat{r}$ este trimisă entității A.
6. Când entitatea A primește $\hat{r}$, va calcula semnătură oarbă $\hat{s} = (\hat{x} \text{ - } \hat{r}x_a) \bmod q$ și o va trimite entității B, care între timp a criptat mesajul ca $\sigma = Enc(m, k_2)$.
7. După ce a primit $\hat{s}$, B va calcula semnătura $s = (\hat{s} + \delta + u) \bmod q$ și va trimite în final criptotextul (σ, r, s, t) entității C.

Designcriptare în care după ce entitatea C a primit criptotextul de la entitatea B, va proceda în felul următor pentru a recupera mesajul și a verifica semnătura făcută de entitatea A pe mesaj:

1. Va calcula cheile $k_1 = t^{-1}g^s{y_a}^r \bmod p$ și $k_2 = k_1^{x_c} \ mod \ p$
2. Va recupera mesajul, $m = Dec(\sigma, k_2)$
3. Apoi va verifica egalitatea $r = KH(m, k_1, t)$. Dacă egalitatea se verifică mesajul primit este acceptat ca fiind de la entitatea A.

Verificarea semnăturii în care dacă entitatea C are nevoie de a demonstra unei entități externe că are o semnătură validă pe mesajul m de la entitatea A, următori pași sunt necesari:
1. Entitatea C va trimite (m, r, s, t) entității externe
2. Entitatea externă va calcula $k_1 = t^{-1}g^s{y_a}^r \ mod \ p$, și va verifica dacă condiția $r = KH(m, k_1, t)$ este adevărată. În caz afirmativ, semnătura este acceptată ca fiind de la entitatea A.

În privința eficienței se poate observa că față de efectuarea operaților exponențiare și de înmulțire, uzuale, apare și o inversiune. Astfel în algoritmul de signcriptare entitatea A împreună cu B, efectuează 5 exponențieri modulare și 3 înmulțiri iar în algoritmul de designcriptare entitatea C efectuează o inversiune, 3 exponențieri modulare și 2 înmulțiri. În cazul proprietăților de securitate, această schemă reușește să asigure confidențialitatea, proprietatea de unforgeability și deasemenea asigură faptul că mesajul nu poate fi văzut de cel care semnează.

Capitolul 6

Aplicație

În această secțiune vom descrie detaliile implementării unei licitații private fără terță parte[20]. Protocolul folosit în implementare a fost prezentat în capitolul anterior, acesta bazându-se pe proprietățile signcriptării inel.

Aplicația a fost implementată în limbajul de programare JAVA. S-a ales această tehnologie deoarece oferă suport nativ pentru operațiile de bază ale acestei aplicații: creare arhitectura Client-Server, creare interfață grafică, lucrul cu elemente criptografice pentru generarea parametrilor protocolului și suport pentru lucrul cu baze de date SQL.

Pentru implementarea operațiilor aplicației biliniare folosite in protocol s-a folosit biblioteca open-source JPBC (Java Pairing Based Cryptography) versiunea 2.0 .

Pentru gestiunea informațiilor despre utilizatori sistemului s-a folosit baza de date SQLite, care necesită zero-configurare, deasemenea o bază de date întreagă este stocat într-un singur fișier și oferă un API, ușor de folosit.

Arhitectura aplicației este Client-Server. Aplicația client poate fi de două tipuri: BuyerApplication și SellerApplication. BuyerApplication este aplicația client la care se conectează cumpărători. Aceasta oferă o interfață grafică prin care cumpărătorul poate posta un proiect, porni o licitație și asigură componetele necesare pentru procesarea ofertelor și verifica rezultatele licitației. SellerApplication este aplicația client, la care se conectează vânzătorii. Aceasta oferă o interfață grafică prin care vânzătorul poate să se înscrie la o licitație și poate plasa oferte pentru o licitație. Serverul asigură conectarea utilizatorilor la aplicație și comunicarea dintre un cumpărător și vânzători. Protocolul de comunicare între client-server este construit peste protocolul TCP. Comunicarea se face prin mesaje care

au următoarea structură: <acțiune><valoare>;

Aplicația dispune de următoarele funcționalități :

- logare utilizator(cumpărător, vânzător) la server, în care serverul va verifica datele de logare. În caz afirmativ acesta va trimite parametri de securitate a sistemului. Aceștia vor fi folosiți de utilizatori pentru ași genera cheia publică și cheia privată ce va fi folosită în signcriptare și designcriptare.

- creare licitație de către cumpărător, în care cumpărătorul poate posta un nou proiect ce va fi distribuit de către server către toți vânzătorii, toate notificările dintre cumpărător și vânzători vor fi afișate într-un bulletin board.

- acceptare/respingere licitație de către vânzător, care va putea să accepte să intre în licitație pentru proiectul primit prin trimiterea unui mesaj către server care va notifica cumpărătorul

- respingere vânzător dintr-o licitație, în care cumpărătorul după ce primește lista de vânzătorii care au acceptat să intre în licitație acesta va putea să respingă o parte din ei. Serverul va notifica vânzători despre acest lucru.

- licitația propriu-zisă, în care după ce în prealabil cumpărătorul a pornit licitația și aceștia au primit notificarea de la server, vânzătorii vor putea plasa oferte. Acestea vor fi trimise, prin server, la cumpărător care le va sorta, va selecta oferta minimă și dacă se confimă valabilitatea ei o va distribui cu ajutorul serverului către vânzători.

- confirmare/infirmare ofertei căștigătoare de către vânzători, după ce aceștia primesc notificare în bulletin board despre oferta câștigătoare aceștia au posibilitatea de a confirma sau infirma faptul că ei au trimis oferta. De exemplu, în cazul confirmării ofertei căștigătoare, se va crea un pachet de confirmare ce va fi trimis la cumpărător. Acesta va verifica acest pachet și dacă acesta este valabil va notifica toți vânzătorii despre căștigător prin bulletin board, păstrând însă anonimitatea acestuia.

Capitolul 7

Concluzii şi direcţii viitoare de cercetare

Lucrarea de față s-a concentrat asupra progresului noi primitive criptografice, *signcriptarea*, prezentând principalele protocoale care au marcat evoluția acestei primitive de la schema originală până la variațiile sale.

Am pornit de la protocolul de bază de signcriptare împreună cu noțiunile sale de securitate. Deasemenea am prezentat două îmbunătățiri a protocolului de baza în privința securității.

În capitolul trei am prezentat protocoalele ce pun baza acestei primitive în criptografia bazată pe identitate și am văzut cum putem să obținem o schemă mai eficientă prin efectuarea unui compromis între nivelul de securitate și eficiența unei scheme.

În capitolul patru am prezentat cele mai importante și cunoscute variații ale signcriptării precum signcriptarea inel, signcriptarea proxy sau signcriptarea partajată. Deasemenea am prezentat diferite aplicații în care aceste variații sunt folosite, spre exemplu MANET, plata electronică sau agenți mobili în comerțul electronic.

În capitolul cinci am demonstrat utilitatea protocoalelor de signcriptare în aplicațiile moderne prin implementarea unei aplicații ce folosește un protocol de licitație privată fără terță parte.

În concluzie pot afirma că signcriptarea oferă îmbunătățiri a eficienței, atât din punct de vedere computațional cât și din punct de vedere a costului comunicării, pentru protocoalele din criptografia cu chei publice care pentru a

asigura securitatea necesită combinarea primitivei criptografice de semnare cu cea de criptare.

Bibliografie

[1] Y. Zheng. Digital signcryption or how to achieve cost (signature & encryption) $\ll$ cost (signature) + cost (encryption). In *Proc. Crypto '97, LNCS 1294, pp. 165-179*, 1997.

[2] F. Bao and R. H. Deng. A Signcryption Scheme with Signature Directly Verifiable by Public Key. *Public Key Cryptography (PKC'98), LNCS 1431, pp. 55-59. Springer-Verlag.* 1998

[3] B. Libert and J.J. Quisquater. Efficient Signcryption with Key Privacy from Gap Diffie-Hellman Groups. *Public Key Cryptography (PKC '04), LNCS 2947, pp. 187-200, Springer-Verlag*, 2004.

[4] M. Barbosa, P. Farshim. Certificateless signcryption. *In: ACM Symposium on Information, Computer and Communications Security-ASIACCS 2008, Tokyo, Japan, pp. 369–372*, 2008.

[5] S.S. Al-Riyami, K.G. Paterson. Certificateless Public Key Cryptography. *In: Laih, C.-S. (ed.) ASIACRYPT 2003. LNCS, vol. 2894, pp. 452–473. Springer, Heidelberg* , 2003.

[6] K. Vivek, S. A. Vivek and S. Ramesh. Elliptic Curve Cryptography. *In: ACM Ubiquity Volume 9 Issue 20, pp. 1-8*, 2008.

[7] X. Boyen. Multipurpose Identity-Based Signcryption: A Swiss Army Knife for Identity-Based Cryptography. *Crypto '03, LNCS 2729, pp. 383-399, Springer-Verlag*, 2003.

[8] P. S. L. M. Barreto, B. Libert, N. McCullagh and J.-J. Quisquater. Efficient and Provably-Secure Identity-Based Signatures and Signcryption from Bilinear Maps. *In: Roy, B. (ed.) ASIACRYPT 2005. LNCS, vol. 3788, pp. 515–532. Springer, Heidelberg*, 2005.

[9] Adi Shamir. How to share a secret. *Communications of the ACM 22 (11): pp. 612–613*, 1979.

[10] Zhang Zhang, Mian Cai and Jin Qu. Signcryption Scheme with Threshold Shared Unsigncryption Preventing Malicious Receivers. *Proceedings of 2002 IEEE Region 10 Conference on Computers, Communications, Control and Power Engineering (TENCON '02), Volume 1, Page(s):196 - 199, 28-31 Oct.*, 2002.

[11] Adi Shamir. Identity based cryptosystems and signature schemes. *Crypto'84, volume 196 of LNCS, pp. 47–53. Springer*, 1984.

[12] J. Malone-Lee. Identity-based signcryption. *Cryptology ePrint Archive, Report 2002/098*, 2002.

[13] L. Guillou and J.-J. Quisquater. A "paradoxical" identity-based signature scheme resulting from zero-knowledge. *Crypto'88, volume 0403 of LNCS, pp. 216–231. Springer*, 1988.

[14] D. Boneh and M. Franklin. Identity-based encryption from the Weil pairing. *PKC'03, volume 2567 of LNCS, pp. 18–30. Springer*, 2003.

[15] R. Sakai and M. Kasahara. ID based cryptosystems with pairing on elliptic curve. *In: 2003 Symposium on Cryptography and Information Security, pp. 22-23, SCIS 2003, Hamamatsu, Japan,*, 2003.

[16] J. K. Liu, J. Baek, and J. Zhou. Online/offline identity-based signcryption revisited. *Cryptology ePrint Archive, Report 2010/274*, 2010.

[17] Z. Xu, G. Dai, D. Yang. An Efficient Online/Offline Signcryption Scheme for MANET. *The Second IEEE International Symposium on Pervasive Computing and Ad Hoc Communications (PCAC-07), pp. 171-176, IEEE Computer Society*, 2007.

[18] T. Singh, R. Ali. An ID-Based Threshold Signcryption Scheme with Threshold Unsigncryption. *International Journal of Advanced Research in Computer Science and Software Engineering (IJARCSSE) 5(6), pp. 307-311*, 2015.

[19] S. S. D. Selvi, S. S. Vivek, S. S. Anand and C. P. Rangan. An identity based ring signcryption scheme with public verifiability. *In Security and Cryptography (SECRYPT), Proceedings of the 2010 International Conference on (pp. 1-10). IEEE*, 2010.

[20] Li, S., Li, X., He, M. X., Zeng, S. K., Tang, X. L. . Sealed-BID electronic auction without the third party. *Wavelet Active Media Technology and Information Processing (ICCWAMTIP), 2014 11th International Computer Conference on. IEEE, 2014. pp. 336-339.*, 2014.

[21] C. Gamage, J. Leiw and Y. Zheng. An Efficient Scheme for Secure Message Transmission using Proxy-Signcryption. *Proceedings of the 22nd Australasian Computer Science Conference, pp. 420-431, Springer*, 1999.

[22] C. Wang, Y. Han, F. Li. A secure mobile agent protocol for M-commerce using self-certified proxy signcryption. *2009 Second International Symposium on Information Science and Engineering, pp. 1431–1446, IEEE* , 2009.

[23] A. Yasmine. New blind signcryption schemes with application to E-cash systems. *International Conference on Computing Communication and Networking Technologies, pp. 1-6, July*, 2014.

[24] A. J. Menezes, P. C. van Oorschot, S. A. Vanstone. Handbook of applied cryptography. *CRC press*, 1996.

[25] J. H. An, T. Rabin. Security for Signcryption: The Two-User Model. *Practical Signcryption Springer Berlin Heidelberg, pp. 43-53* , 2010.

[26] N.P. Smart, F. Vercauteren. On computable isomorphisms in efficient pairing based systems. *Cryptology ePrint Archive, Report 2005/116* , 2016, http://eprint.iacr.org/2005/116.

[27] D. Chaum. Blind signature systems. *In Advances in Cryptology — CRYPTO '83, page 153. Plenum Press*, 1984.