UNIVERSITATEA DIN BUCUREŞTI
FACULTATEA DE MATEMATICĂ ŞI INFORMATICĂ

Protocoale de stabilire a cheilor de grup bazate pe scheme de partajare a secretelor (Secret Sharing-based Group Key Establishment)

LUCRARE DE DOCTORAT
REZUMAT

Coordonator:
Prof.univ.dr.Adrian ATANASIU

Doctorand:
Ruxandra-Florentina OLIMID

Bucureşti
2013

Informaţii de contact ale autorului:
ruxandra.olimid@fmi.unibuc.ro

Comisia de examinare:
Conf. Dr. Radu Gramatovici (Universitatea din Bucureşti, Romania - preşedinte)
Prof. Dr. Adrian Atanasiu (Universitatea din Bucureşti, Romania - conducător ştiinţific)
Prof. Dr. Victor Valeriu Patriciu (Academia Tehnică Militară, Romania - referent naţional)
Prof. Dr. Ferucio Laurenţiu Ţiplea (Universitatea din Iaşi, Romania - referent naţional)
Prof. Dr. Bogdan Warinschi (Universitatea din Bristol, Anglia - referent internaţional)

Abstract

Aplicaţiile de grup permit mai multor utilizatori să partajeze resurse sau să desfăşoare activităţi în colaborare, oferind posibilitatea unor drepturi şi responsabilităţi diferenţiate în cadrul grupului. Securitatea reprezintă un aspect important pentru astfel de aplicaţii, fiind dificil de implementat, mai ales când dimensiunea grupului este mare şi membrii acestuia sunt localizaţi în zone geografice sau de reţea diferite şi beneficiază de mecanisme de protecţie diverse. Pentru a dobândi proprietăţi criptografice de bază (precum confidenţialitate, autenticitate şi integritate) de obicei este necesar ca membrii grupului să stabilească o cheie secretă comună. Aceasta se obţine în urma execuţiei unui *protocol de stabilire a cheilor de grup*.

Teza se rezumă la studiul protocoalelor de stabilire a cheilor de grup bazate pe *scheme de partajare a secretelor*, un mecanism criptografic care permite divizarea unui secret în mai multe componente astfel încât numai submulţimi autorizate de astfel de componente să permită reconstrucţia. Deşi partajarea secretelor aduce multiple avantaje când este utilizată în cadrul protocoalelor de stabilire a cheilor de grup, există în momentul de faţă două deficienţe principale: (1) multe protocoale nesigure au fost publicate în ultimii ani şi (2) foarte puţine construcţii se bazează pe demonstraţii riguroase de securitate.

Prima parte a tezei se concentrează pe scheme de partajare a secretelor. Rezumă o abordare non-clasică a acestora, defineşte o nouă schemă vizuală de partajare şi analizează posibilitatea construcţiei maliţioase a sistemelor de partajare. A doua parte se axează pe protocoale de stabilire a cheilor de grup care utilizează scheme de partajare a secretelor. Introduce multiple atacuri asupra protocoalelor recent publicate, subliniind astfel necesitatea demonstaţiilor de securitate. Rezumă proprietăţile necesare pentru atingerea unui nivel de securitate optim şi analizează pe scurt metode de analiză a securităţii. În final, introduce un protocol de punere de acord asupra cheii comune de grup care atinge un nivel ridicat de securitate în timp ce menţine constant numărul de runde de comunicaţie (indiferent de dimensiunea grupului).

Tuturor celor care mi-au influenţat viaţa şi munca.

Cuprins

Lista de simboluri şi notaţii **7**

Prezentare generală şi organizare **8**

Partea I. Partajarea secretelor **11**

1 Introducere în partajarea secretelor **13**

2 Scheme de partajare a secretelor **15**
2.1 Scheme de m-partajare 15
2.2 O nouă schemă vizuală de partajare 16

3 SETUP asupra SSS care utilizează generatoare de numere aleatoare **21**

Partea II. Protocoale de stabilire a cheilor de grup bazate pe scheme de partajare a secretelor **25**

4 Introducere în protocoale de stabilire a cheilor de grup **27**

5 Protocoale GKE informal sigure bazate pe SSS **29**
5.1 Harn şi Lin (2010) 29
5.2 Hsu et al. (2012) 29
5.3 Sun et al. (2012) 33
5.4 Yuan et al. (2013) 33

6 Protocoale GKE formal sigure bazate pe SSS **43**
6.1 Un nou protocol 43
6.2 Demonstraţii de securitate 43
6.3 Analiză 45

Concluzii şi direcţii viitoare de cercetare **47**

Bibliografie **49**

Lista de simboluri şi notaţii

$\mathbb{N}, \mathbb{Z}, \mathbb{R}$	=	mulţimea numerelor naturale, întregi, respectiv reale
p, q	=	numere naturale prime
$\mathbb{Z}_p$	=	mulţimea întregilor modulo $p \in \mathbb{N}$, i.e. {0, ..., p-1}
$\mathbb{Z}_p^*$	=	grupul multiplicativ {1, ..., p-1}
m	=	numărul posibil de utilizatori (participanţi / membri ai grupului / entităţi / jucători / părţi)
$\mathcal{U} = \{U_1, \ldots, U_m\}$	=	mulţimea utilizatorilor posibili
(s_i)	=	sesiunea i (a unui protocol)
t	=	numărul participanţilor la o sesiune dată
$\mathcal{U}_{(s_i)}$	=	mulţimea participanţilor la sesiunea (s_i)
k, $k_{(s_i)}$	=	cheia de grup, cheia de grup a sesiunii (s_i)
$\mathcal{D}$	=	dealerul
$\mathcal{AS}$	=	structura de acces a unei SSS
S	=	secretul partajat
S^i	=	al i-lea secret partajat
$Sp(S)$	=	mulţimea tuturor secretelor posibile
s_i, s_i^j	=	componenta i a unui secret S, componenta i a unui secret S^j
$Sp(s)$	=	mulţimea tuturor componentelor posibile
$S(x, y)$	=	pixelul (x, y) al imaginii secrete S (în cazul VSSS)
$s_i(x, y)$	=	pixelul (x, y) al componentei s_i (în cazul VSSS)
$x \leftarrow^R X$	=	o alegere uniform aleatoare a lui x din mulţimea X
$A \rightarrow B : M$	=	un mesaj M transmis de A lui B
$A \rightarrow^* : M$	=	un mesaj M de tip broadcast transmis de A
$\mathsf{H}, \mathsf{H}_i, h, h_i$	=	funcţii hash, $i \in \mathbb{N}$
(pk, sk)	=	o pereche cheie publică - cheie privată
(pk_i, sk_i)	=	o pereche cheie publică - cheie privată a utilizatorului U_i
$\Sigma.\mathsf{Sign}_{U_i}(M)$	=	semnătura utilizatorului U_i asupra mesajului M
$\Sigma.\mathsf{Verify}_{U_i}(M, \sigma)$	=	verificarea semnăturii σ a utilizatorului U_i corespunzătoare mesajului M
$\mathcal{K}$	=	parametrul de securitate
q_r	=	numărul maxim de interogări a unei funcţii hash
q_s	=	numărul mamxim de sesiuni (concurente)
$Pr[\mathsf{X}]$	=	probabilitatea de realizare a evenimentului X
$\mathsf{Adv}_{\mathcal{A}}^{\{\mathsf{AKE},\mathsf{MA},\mathsf{Con}\}}$	=	avantajul adversarului PPT (Probabilistic Polynomial Time) $\mathcal{A}$ împotriva securităţii AKE, securităţii MA, respectiv contributivităţii unui protocol

Prezentare generală şi organizare

Partea I se concentrează asupra schemelor de partajare a secretelor, dar se rezumă strict la noţiunile şi schemele utilizate pe parcursul lucrării şi la contribuţiile personale. Conţine următoarele capitole:

- **Capitolul 1** introduce schemele de partajare şi clasificarea acestora. Scopul principal constă în familiarizarea cititorului cu noţiunile necesare pentru parcurgerea tezei, fără să aibă ca obiectiv prezentarea unui studiu complet.
- **Capitolul 2** descrie mai întâi două scheme clasice de partajare a secretelor, care ulterior vor fi utilizate în construcţia protocoalelor de stabilire a cheilor de grup. Apoi, prezintă perspectiva non-clasică a *schemelor de m-partajare*, o abordare care aduce avantaje suplimentare în cadrul stabilirii cheilor de grup. În final, capitolul introduce o schemă vizuală de partajare a secretelor, analizează proprietăţile acesteia şi prezintă rezultatele obţinute în urma implementării.
- **Capitolul 3** extinde SETUP (Secretly Embedded Trapdoor with Universal Protection) la schemele de partajare, arătând vulnerabilitatea acestora în cazul în care utilizează o cantitate suficientă de informaţie aleatoare. Analizează atacul propus şi consideră anumite tehnici de protecţie. Cu scopul de a exemplifica utilitatea practică, mecanismul este particularizat pentru schema de partajare Shamir, cea mai populară în literatura de specialitate.

Partea II este centrată asupra protocoalelor de stabilire a cheilor de grup bazate pe partajarea secretelor. În funcţie de modul de analiză a securităţii, protocoalele se împart în: protocoale care beneficiază de o demonstraţie formală (riguroasă) de securitate şi protocoale care se bazează numai pe argumente informale şi incomplete. Conţine următoarele capitole:

- **Capitolul 4** introduce protocoalele de stabilire a cheilor de grup şi clasificarea acestora în protocoale de transfer şi protocoale de punere de acord (agreare). Crează legătura cu schemele de partajare, utilizate în construcţia acestor protocoale.
- **Capitolul 5** se centrează pe protocoalele care nu deţin o demonstraţie riguroasă de securitate. Tratează patru astfel de protocoale introduse în ultimii ani (2010-2013), descrie mai multe atacuri asupra acestora şi indică posibile metode de îmbunătăţire.
- **Capitolul 6** se concentrează asupra protocoalelor care conţin o demonstraţie riguroasă de securitate. Mai întâi, descrie noţiunile de bază utilizate în demonstraţiile formale,

tehnica *succesiunii de jocuri (sequence of games)*, modelul GBG şi limitările modelelor de securitate actuale. Apoi, prezintă două protocoale existente care beneficiază de o demonstraţie formală de securitate. Contribuţia principală a capitolului constă în introducerea unui nou protocol demonstrat sigur în modelul GBG, care menţine acelaşi număr de runde ca protocoalele existente, dar beneficiază de multiple avantaje datorate schemelor de m-partajare.

În varianta completă a tezei, secţiunea **Contribuţii personale** a fiecărui capitol sumarizează rezultatele proprii şi indică lucrările originale (excepţie fac Capitolele 1 şi 4, care sunt introductive). Rezumatul ignoră această secţiune pentru că se axează pe contribuţiile personale, reducând la minim prezentarea noţiunilor deja existente.

În finalul lucrării, secţiunea **Concluzii şi direcţii viitoare de cercetare** reia rezultatele personale prezentate şi sugerează posibile direcţii viitoare de studiu.

Lista de Simboluri şi Notaţii indică notaţiile utilizate pe parcursul lucrării. Deşi câteodată amintim cititorului să facă referire la această listă, de obicei nu repetăm semnificaţia simbolurilor în cadrul lucrării. Ca o remarcă generală, omitem în general să specificăm spaţiul matematic în care se lucrează, acesta fiind evident din descriere (de exemplu nu menţionăm mod p de fiecare dată când operaţiile au loc într-un grup finit de ordin p).

Varianta completă a tezei conţine două **Anexe**, care rezumă cunoştiinţele necesare de criptografie şi matematică:

- **Anexa A. Noţiuni de bază în Criptografie** descrie primitivele criptografice utilizate pe parcursul lucrării. Considerăm că cititorul este deja familiarizat cu acestea, însă le reamintim pentru completitudine şi specificarea notaţiilor.
- **Anexa B. Noţiuni de bază în Matematică** prezintă câteva concepte şi leme care se aplică în cadrul tezei.

Partea I

Partajarea secretelor

Capitolul 1

Introducere în partajarea secretelor

Definiţia 1.1. ***(Scheme de partajare a secretelor (SSS[1]))****. O schemă de partajare a secretelor este un proces sau un protocol care împarte un secret primit la intrare în componente astfel încât doar mulţimi autorizate de componente permit reconstrucţia secretului iniţial [8].*

Mai formal, este o pereche de algoritmi $(\mathsf{Share}, \mathsf{Rec})$*, unde:*

1. $\mathsf{Share}(S, m)$ *este un algoritm nedeterminist de partajare care primeşte la intrare un secret* S *şi scoate la ieşire o mulţime de* m *componente* $\{s_1, \ldots, s_m\}$*;*

2. $\mathsf{Rec}(s_{i_1}, \ldots, s_{i_t})$*,* $t \leq m$ *este un algoritm determinist de reconstrucţie din componente care scoate la ieşire* S *dacă* $\{s_{i_1}, \ldots, s_{i_t}\}$ *reprezintă o mulţime autorizată.*

Mulţimea tuturor mulţimilor autorizate se numeşte *structură de acces* şi se notează cu $\mathcal{AS}$. În mod normal, componentele sunt distribuite mai multor participanţi, astfel încât ne vom referi la o mulţime *autorizată* de participanţi ca fiind o mulţime capabilă să determine secretul prin colaborare. O structură de acces se numeşte *monotonă* dacă orice mulţime care conţine o mulţime autorizată este de asemenea autorizată.

În general, SSS constau în mai multe faze:

1. *Iniţializare.* Defineşte mediul de desfăşurare al schemei: parametrii, spaţiul secretelor şi al componentelor, alte premise.

2. *Partajarea.* Descrie algoritmul de divizare al secretului în componente. În multe cazuri, o entitate de încredere numită *dealer* realizează partajarea.

3. *Distribuţia.* Indică componentele transmise fiecărui participant prin canale sigure. În general, fiecare participant primeşte o singură componentă. Există însă cazuri când o entitate primeşte mai multe componente (şi deci capătă o putere mai mare în cadrul grupului) sau chiar o întreagă mulţime autorizată (şi deci devine capabilă să determine secretul fără colaborarea cu alţi membri ai grupului[2]).

4. *Reconstrucţia.* Explicitează formulele sau algoritmii necesari pentru determinarea secretului dintr-o mulţime autorizată de componente.

[1] Secret Sharing Schemes.

[2] Acesta este cazul schemelor de m-partajare.

Definiţia 1.2. ***(Scheme de partajare perfecte)*** [8]. O schemă de partajare este *perfectă* dacă componentele corespunzătoare mulţimilor neautorizate nu oferă nici o informaţie despre secret.

Definiţia 1.3. ***(Scheme de partajare ideale)*** [8]. O schemă de partajare este ideală dacă este perfectă şi dimensiunea componentelor este egală cu dimensiunea secretului.

Menţionăm două tipuri de SSS cu structuri de acces speciale:

- *SSS cu structura de acces de tip prag.* Structura de acces conţine toate mulţimile cu cardinalul cel puţin t, $1 \leq t \leq m$. Informal, orice t sau mai multe componente sunt suficiente pentru reconstrucţia secretului, în timp ce mai puţin de t componente nu. Numim o astfel de schemă (t, m)-schemă de partajare.
- *SSS Unanime.* Structura de acces conţine ca singur element mulţimea tuturor componentelor. Informal, toate componentele sunt necesare pentru reconstrucţia secretului.

În funcţie de secretul partajat, SSS se diferenţiază în:

- *Scheme text.* Secretul partajat este o secvenţă de biţi (ex.: un element al unui corp finit sau un text). Sunt cele mai populare scheme de partajare şi prezintă o utilitate crescută în practică.
- *Scheme vizuale.* Secretul partajat este o imagine.
- *Scheme audio sau video.* Secretul partajat este un fişier audio sau video.

Deşi există multiple alte clasificări ale SSS (ex.: dinamice vs. statice, care partajează un singur secret vs. care partajează secrete multiple, proactive vs. reactive, cu dealer vs. fară dealer, etc.) şi proprietăţi speciale (ex.: detecţia şi identificarea trişorilor, verificabilitate) ne rezumăm strict la noţiunile necesare pentru lucrarea de faţă.

Capitolul 2

Scheme de partajare a secretelor

2.1 Scheme de m-partajare

Definiţia 2.1. ***(Scheme de m-partajare).*** *O schemă de m-partajare este o schemă care împarte un secret de m ori în acelaşi număr de componente t utilizând aceeaşi schemă de partajare de bază (i.e. aceiaşi algoritmi* Share *şi* Rec*).*

Altfel spus, o *schemă de m-partajare* execută de m ori o schemă de bază pentru acelaşi secret S şi acelaşi număr de componente t. Fie $\mathcal{AS}$ structura de acces a schemei de m-partajare şi $\mathcal{AS}_i$, $i = 1, \ldots, m$ structurile de acces corespunzătoare execuţiilor schemei de bază. Se observă imediat că:

$$\mathcal{AS}_1 \cup \cdots \cup \mathcal{AS}_n \subseteq \mathcal{AS}. \tag{2.1}$$

Definiţia 2.2. ***(Scheme de m-partajare perfecte)*** *[20]. O schemă de m-partajare este perfectă dacă se satisfac următoarele condiţii: (1) structura sa de acces este dată de $\mathcal{AS} = \mathcal{AS}_1 \cup \cdots \cup \mathcal{AS}_n$; (2) mulţimile neautorizate nu oferă nici o informaţie despre secret.*

Iniţializare. Fie $m = 2, t = 2, q \geq 3$ un număr prim (mare) şi $S \in \mathbb{Z}_q$ secretul;

Partajare 1. Dealerul $\mathcal{D}$:
1.1. alege $s_1^1 \leftarrow^R \mathbb{Z}_q$;
1.2. calculează $s_2^1 = S - s_1^1$;

Distribuţie 1. Dealerul $\mathcal{D}$:
2.1. transmite:
$\mathcal{D} \rightarrow U_1 : s_i^1$, $i = 1, 2$;

Reconstrucţie 1. Utilizatorul U_1:
3.1. calculează $S = s_1^1 + s_2^1$;

Partajare 2. Dealerul $\mathcal{D}$:
1.1. alege $s_1^2 \leftarrow^R \mathbb{Z}_q$;
1.2. calculează $s_2^2 = S - s_1^2$;

Distribuţie 2. Dealerul $\mathcal{D}$:
2.1. transmite:
$\mathcal{D} \rightarrow U_2 : s_i^2$, $i = 1, 2$;

Reconstrucţie 2. Utilizatorul U_2:
3.2. calculează $S = s_1^2 + s_2^2$;

Figura 2.1: O schemă de 2-partajare perfectă bazată pe schema Karnin et al. [20]

Fig.2.1 exemplifică o schemă de 2-partajare perfectă bazată pe schema de partajare prezentată de Karnin et al. [6] şi utilizată în cazul general (m partajări) de Sun et al. [23].

2.2 O nouă schemă vizuală de partajare

Am introdus în 2010 o schemă de partajare vizuală care permite împărţirea unei imagini alb-negru în componente color [11]. Fig.2.3 descrie schema în detaliu.

Imaginile sunt reprezentate de mulţimi de pixeli indicaţi de coordonatele lor ($S(x, y)$ este un pixel al imaginii secrete şi $s(x, y)$ un pixel al componentei s) şi definiţi prin culoare, considerată în modelul RGB. Fig.2.2 ilustează acest model. Fiecare culoare este codată ca un triplet de octeţi care reprezintă proporţia de apariţie a celor 3 culori primare; listăm cele mai importante valori în Tabelul 2.1.

Fig.2.4 exemplifică o mulţime posibilă de componente în cazul partajării unei imagini de 2x2 pixeli albi, respectiv negri pentru $m = 4$ participanţi.

Schema poate fi utilizată numai pentru $m \geq 3$, pentru că cel puţin 3 componente sunt necesare pentru partajarea unui pixel alb (un pixel corespunzător R, unul G şi unul B în componente distincte).

Analizăm cantitatea de informaţie dezvăluită prin cooperarea a t participanţi, $1 \leq t \leq m$:

- $t \leq 2$. Cel mult doi participanţi nu deţin nici o informaţie, pentru ca cel puţin 3 componente sunt necesare pentru obţinerea unui pixel alb. Aşadar, ei pot reconstrui doar o imagine complet neagră.

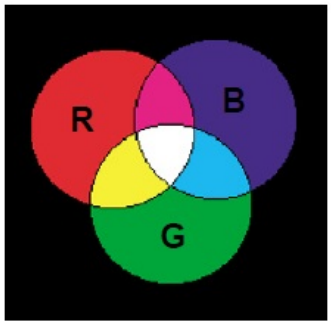

Figura 2.2: Modelul RGB

Tabelul 2.1: Codarea RGB

Culoare	Codare	Culoare	Codare
Alb	$W = (255, 255, 255)$	Negru	$Bl = (0, 0, 0)$
Roşu	$R = (255, 0, 0)$	Galben	$Y = (255, 255, 0)$
Verde	$G = (0, 255, 0)$	Magenta	$M = (255, 0, 255)$
Albastru	$B = (0, 0, 255)$	Cyan	$C = (0, 255, 255)$

Iniţializare. Fie S imaginea secretă;

Partajare. Dealerul $\mathcal{D}$, pentru toţi pixelii $S(x, y)$:
1.1. dacă $S(x, y) = W$:
1.1.1. alege $s_i(x, y) \leftarrow^R \{R, G, B\}$, $i = 1, \ldots, m$;
1.1.2. alege $i_1, i_2, i_3 \leftarrow^R \{1, \ldots, m\}$ distincte;
1.1.3. $s_{i_1} \leftarrow R, s_{i_2} \leftarrow G, s_{i_3} \leftarrow B$;
1.2. dacă $S(x, y) = Bl$:
1.2.1. alege $X \leftarrow^R \{\{R, G\}, \{R, B\}, \{G, B\}\}$;
1.2.2. alege $s_i(x, y) \leftarrow^R X$, $i = 1, \ldots, m$;
1.2.3. alege $i_1, i_2 \leftarrow^R \{1, \ldots, m\}$ distincte;
1.2.4. $s_{i_1} \leftarrow^R X, s_{i_2} \leftarrow X \setminus \{s_{i_1}\}$;

Distribuţie. Dealerul $\mathcal{D}$:
2.1. transmite (prin canale de comunicaţie sigure):
$\mathcal{D} \rightarrow U_i : s_i$, $i = 1, \ldots, m$;

Reconstrucţie. Cei m participanţi:
3.1. refac imaginea secretă S, unde:
3.1.1. $S(x, y) = W$ dacă există 3 pixeli corespunzători $s_i(x, y)$ distinct coloraţi, $i = 1, \ldots, m$;
3.1.2. $S(x, y) = Bl$ dacă există numai 2 pixeli corespunzători $s_i(x, y)$ distinct coloraţi $i = 1, \ldots, m$.

Figura 2.3: Schema de partajare vizuală Olimid [11]

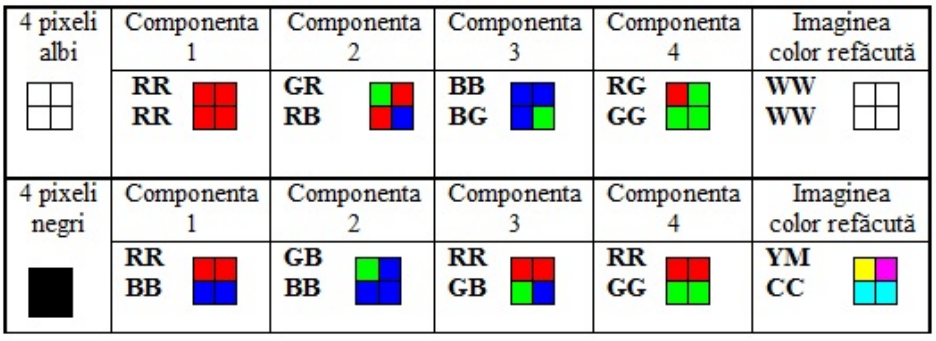

Figura 2.4: O posibilă partajare pentru 4 pixeli albi şi 4 pixeli negri ($m = 4$) [12]

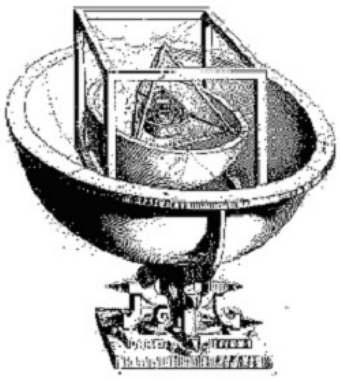

Figura 2.5: Imaginea secretă de test

- $2 < t < m$. Mai mult de 2 participanţi obţin informaţie suplimentară despre secret: un pixel alb obţinut prin reconstrucţie corespunde cu siguranţă unui pixel alb din imaginea iniţială; nu se poate afirma însă nimic despre pixelii negri obţinuţi prin reconstrucţie. Probabilitatea de reconstrucţie corectă a unui pixel creşte cu numărul t de participanţi.
- $t = m$. Toţi participanţii determină imaginea partajată prin punerea în comun a componentelor pe care le deţin.

Am implementat schema propusă în limbajul de programare Python, utilizând IDLE (Integrated DeveLopment Environment 2.6.6) ca mediu de dezvoltare [24]. Un motiv pentru care am ales Python este că pune la dispoziţia dezvoltatorilor în variantă gratuită o librărie de lucru cu imagini numită PIL (Python Image Library) care permite: definirea imaginilor în diferite moduri (alb-negru, color conform modelului RBG), manipularea imaginilor la nivel de pixel, a benzilor de culoare (R, G şi B), etc. Menţionăm că pentru generarea aleatoare a valorilor am utilizat de asemenea modulul *random* existent în Python.

Fig.2.5 indică imaginea iniţială. Fig.2.6 exemplifică un set de componente posibile pentru $m = 4$. Fig.2.7 indică posibile reconstrucţii din 2 sau 3 componente şi imaginea rezultată din utilizarea tuturor componentelor.

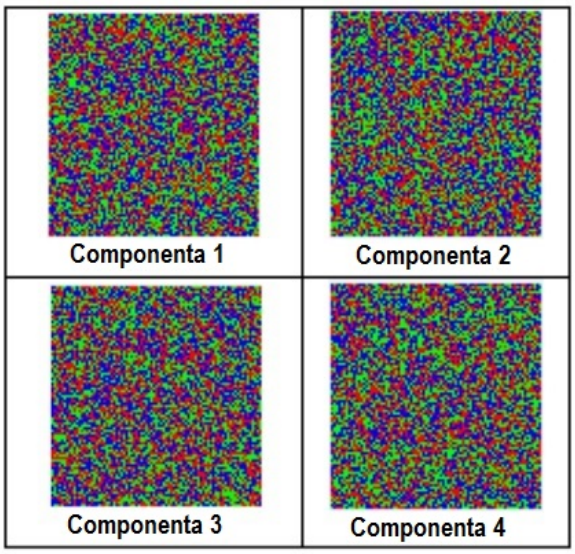

Figura 2.6: O posibilă partajare ($m = 4$) [11]

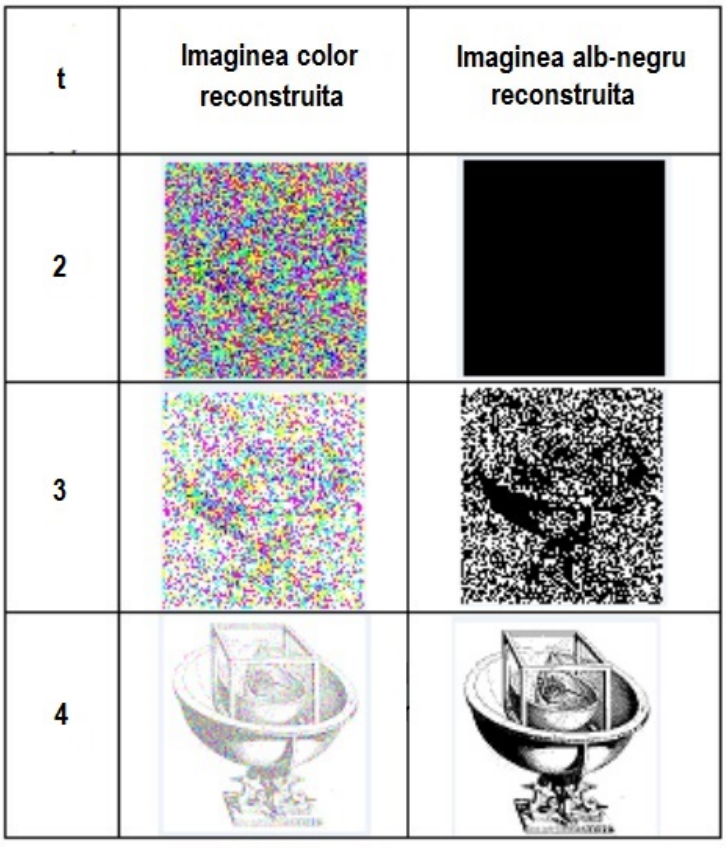

Figura 2.7: Exemple de imagini obţinute prin reconstrucţie ($m = 4$) [11]

Capitolul 3

SETUP asupra SSS care utilizează generatoare de numere aleatoare

Mecanismul SETUP (Secretly Embedded Trapdoor with Universal Protection) a fost definit de Young şi Yung [26]. Acesta reprezintă o tehnică maliţioasă realizată de către producătorul unui sistem criptografic care constă în implementarea unui canal care permite scurgerea de informaţie secretă criptată. Criptarea se realizează folosind cheia publică a atacatorului, acesta fiind deci singurul care are acces la informaţie.

Implementarea este de tip *cutie neagră* (*black box*): un utilizator are acces numai la valorile de intrare şi ieşire ale sistemului, în timp ce algoritmii şi memoria internă rămân inaccesibile. Modelul permite astfel atacatorului să schimbe construcţia internă, în timp ce utilizatorii nu pot suspecta nici un comportament maliţios dacă nu sunt aparent afectate valorile de intrare şi ieşire.

Definiţia 3.1. ***(Secretly Embedded Trapdoor with Universal Protection (SETUP))*** [27]. Fie C un criptosistem de tip cutie neagră cu o specificaţie publică. Un mecanism de tip SETUP (Secretly Embedded Trapdoor with Universal Protection) este o modificare la nivel algoritmic al lui C cu obţinerea lui C' astfel încât:

1. Intrarea lui C' este conformă cu specificaţiile publice ale intrării lui C;
2. C' criptează în mod eficient utilizând cheia publică a atacatorului, memorată în C';
3. Cheia privată a atacatorului nu este conţinută în C' şi este cunoscută numai de către atacator.
4. Ieşirea lui C' este conformă cu specificaţiile publice ale ieşirii lui C;
5. Ieşirile lui C şi C' sunt polinomial indistinctibile pentru oricine cu excepţia atacatorului;
6. După descoperirea atacului SETUP în implementare (de exemplu prin reverse-engineering) utilizatorii (cu excepţia atacatorului) nu pot determina chei anterioare (ideal, nici chei viitoare).

Un criptosistem modificat care implementează SETUP se numeşte *contaminat* [26].

De la introducerea sa, SETUP a fost aplicat diferitelor sisteme de criptare, scheme de semnătură electronică, algoritmi de generare de chei, sisteme de vot electronic. Introducem acum o metodă generală de implementare în cadrul SSS care utilizează suficientă informaţie aleatoare [14]. Scopul principal este acela de a oferi atacatorului un avantaj semnificativ pentru reconstrucţia secretului, fără să necesite acces la o mulţime autorizată de componente.

Atacul SETUP propus devine posibil în următoarele ipoteze:

1. Mecanismul de partajare este implementat ca o cutie neagră care poate stoca într-o memorie non-volatilă informaţie din mai multe execuţii ale SSS;
2. Mecanismul de partajare generează o mulţime de valori aleatoare care sunt ulterior utilizate pentru calculul componentelor (ambele operaţii se realizează în cadrul cutiei negre);
3. Numărul de valori aleatoare din fiecare componentă este mai mare sau egal cu numărul de componente ale secretului (sau o componentă poate fi considerată un număr aleator);
4. Componentele sunt distribuite prin canale securizate;
5. Atacatorul este întotdeauna unul dintre participanţi;
6. Mai multe secrete sunt partajate.

Fig.3.1 descrie algoritmul intern al sistemului contaminat şi Fig.3.2 explică recontrucţia secretului aşa cum este realizată de atacator. Sistemul de criptare cu cheie publică este ales astfel încât: (1) să fie Ind-CPA sigur; (2) să accepte ca text criptat valid orice valoare din $Sp(s)^u$, unde $u \geq 1$, fixat; (3) $u \in \mathbb{Z}^*$ este minim. Viteza crescută de criptare constituie de asemenea un avantaj.

Teorema 3.1. [14]. Varianta contaminată a SSS satisface proprietatea de indistinctibilitate faţă de schema originală.

Teorema 3.2. [14] Varianta contaminată a SSS menţine securitatea schemei originale pentru oricine cu excepţia deţinătorului cheii private.

Exemplificăm aplicabilitatea tehnicii propuse asupra schemei Shamir, cea mai cunoscută şi utilizată SSS din literatură [14]. Schema este ideală, deci atacatorul nu poate desfăşura atacul fără a cunoaşte cel puţin o altă componentă ($u > 1$ pentru că nu există un sistem de criptare cu cheie publică Ind-CPA sigur pentru care dimensiunea textului criptat să fie egală cu cea a textului clar). Soluţia indicată este optimă: atacatorul necesită cunoaşterea unei singure componente suplimentare. Fig.3.3 descrie algoritmul contaminat al schemei Shamir şi Fig.3.4 explică procesul de reconstrucţie urmat de atacator.

Intrare: m numărul de participanţi, $S^1, S^2, \ldots$ secretele care se partajează;
Ieşire: $s_1^i, s_2^i, \ldots, s_m^i$ componentele corespunzătoare secretului S^i, $i \geq 1$;

1: **if** i==1 **then**
2: $\alpha \leftarrow^R Sp(s)$;
3: $s_1^1 = \alpha$;
4: $\mathsf{H}(ID||s_1^1)$ este stocat în memoria non-volatilă;
5: $s_2^1, s_3^1, ..., s_m^1$ sunt calculate conform cu schema originală, luând în considerare componenta s_1^1;
6: **else**
7: $(s_1^i, ..., s_u^i) = \mathsf{Enc}(pk, S^i \otimes \mathsf{H}(ID||s_1^{i-1}))$, $1 \leq u \leq m$;
8: $\mathsf{H}(ID||s_1^{i-1})$ este înlocuit în memoria non-volatilă cu $\mathsf{H}(ID||s_1^i)$;
9: $s_{u+1}^i, s_{u+2}^i, ..., s_m^i$ sunt calculate conform cu schema originală, luând în considerare componentele $s_1^i, ..., s_u^i$.
10: **end if**

Figura 3.1: Atac SETUP asupra schemelor de partajare care utilizează valori aleatoare [14]

Pas 1. U_1 memorează componenta sa s_1^{i-1} din execuţia anterioară;

Pas 2. U_1 foloseşte propria componentă s_1^i şi $s_2^i, ..., s_u^i$ pentru a calcula secretul S^i:
$S^i = \mathsf{Dec}(sk, (s_1^i, ..., s_u^i)) \otimes (\mathsf{H}(ID||s_1^{i-1}))^{-1}$.

Figura 3.2: Avantajul atacatorului în determinarea secretului în SSS contaminată [14]

Input: m numărul de participanţi, $S^1, S^2, \ldots$ secretele care se partajează, $q \geq m+1$ prim;
Output: $s_1^i, s_2^i, \ldots, s_m^i$ componentele corespunzătoare secretului S^i, $i \geq 1$;

1: **if** i==1 **then**
2: $x_j \leftarrow^R \mathbb{Z}_q$, $j = 1, \ldots, m$;
3: $s_1^1 \leftarrow^R \mathbb{Z}_q$;
4: este generat un polinom de grad $t-1$ a.î. $f^1(x) = S^1 + a_1^1 x + \ldots. + a_{t-1}^1 x^{t-1}$ şi $f(x_1) = s_1^1$;
5: $s_j^1 = f^1(x_j)$, $j = 2, \ldots, m$;
6: $H(ID||s_1^1)$ este stocat în memoria non-volatilă;
7: **else**
8: $(s_1^i, s_2^i) = \mathsf{Enc}(pk, S^i \cdot \mathsf{H}(ID||s_1^{i-1}))$;
9: $\mathsf{H}(ID||s_1^{i-1})$ este înlocuit în memoria non-volatilă cu $\mathsf{H}(ID||s_1^i)$;
10: $a_j^i \leftarrow^R \mathbb{Z}_q$, $j = 3, \ldots, t-1$;
11: a_1^i şi a_2^i sunt calculate ca soluţie a sistemului:

$$\begin{cases} s_1^i &= f^i(x_1) \\ s_2^i &= f^i(x_2) \end{cases}$$

unde $f^i(x) = S^i + a_1^i x + \cdots + a_{t-1}^i x^{t-1}$ este un polinom de grad $t-1$;
12: $s_j^i = f^i(x_j)$.
13: **end if**

Figura 3.3: Atac SETUP asupra schemei de partajare Shamir [14]

Pas 1. U_1 memorează componenta sa s_1^{i-1} din execuţia anterioară;

Pas 2. U_1 foloseşte propria componentă s_1^i şi s_2^i pentru a calcula secretul S^i:
$S^i = \mathsf{Dec}(sk, (s_1^i, s_2^i)) \cdot (\mathsf{H}(ID||s_1^{i-1}))^{-1}$.

Figura 3.4: Avantajul atacatorului în determinarea secretului în SSS Shamir contaminată [14]

Partea II

Protocoale de stabilire a cheilor de grup bazate pe scheme de partajare a secretelor

Capitolul 4

Introducere în protocoale de stabilire a cheilor de grup

Definiţia 4.1. ***(Protocoale de stabilire a cheilor de grup (GKE[1]))*** [8]. Un *protocol de stabilire a cheilor de grup (GKE)* este un proces sau un protocol prin care un secret devine comun mai multor părţi, în scopul unei utilizări criptografice ulterioare.

Scopul principal al unui protocol GKE este acela de a stabili o cheie comună între membrii autorizaţi ai unui grup, fără a scurge informaţie altor entităţi. Participanţii *autorizaţi* se mai numesc *calificati*, *legitimi* sau *privilegiaţi*. Un protocol se execută de mai multe ori, fiecare execuţie reprezentând o *sesiune*, identificată în mod unic de un identificator (**sid**[2]). Numim *cheie de sesiune* un secret comun rezultat în urma unei execuţii a protocolului. Aceasta persistă pentru o scurtă perioadă de timp, o prezumţie naturală în criptografie. Pentru a deveni eligibil să ia parte la o sesiune, un utilizator trebuie ca în prealabil să se înregistreze în cadrul grupului, operaţie în urma căreia dobândeşte o *cheie de lungă durată* pe care o va utiliza pe parcursul derulării protocolului.

În general, protocoalele GKE constau în mai multe faze:

1. *Iniţializarea.* Defineşte mediul de desfăşurare al protocolului: parametrii, spaţiul cheilor posibile, alte premise.

2. *Înregistrarea.* În funcţie de scenariu, după înregistrare, un utilizator poate partaja o cheie secretă (sau o parolă) cu o autoritate de încredere (KGC[3]) sau poate genera o pereche cheie publică - cheie privată pentru semnături ulterioare.

3. *Execuţia.* Descrie algoritmul criptografic, inclusiv operaţiile executate şi mesajele transmise. De obicei constă în mai multe runde de comunicaţie între participanţi.

4. *Calculul cheii de grup.* Explicitează algoritmii necesari pentru obţinerea cheii din cunoştinţele dobândite după faza de execuţie. Este câteodată integrată ca o rundă în faza de execuţie.

[1] Group Key Establishment.
[2] Session id.
[3] Key Generation Center.

5. *Confirmarea cheii de grup.* Asigură că toate entităţile autorizate deţin într-adevăr cheia privată comună şi nimeni în afara lor nu are acces la aceasta. Este o fază opţională care în general apare din motive de securitate.

Protocoalele GKE se împart în două clase: *protocoale de transfer (GKT*[4]*)* şi *protocoale de punere de acord (GKA*[5]*).*

Definiţia 4.2. ***(Protocoale de transfer al cheilor de grup(GKT))*** [8]. Un *protocol de transfer al cheilor de grup (GKT)* este un protocol de stabilire a cheilor de grup în care o entitate generează (sau obţine prin alte mijloace) o cheie şi o transferă prin canale sigure celorlalte entităţi autorizate.

Definiţia 4.3. ***(Protocoale de punere de acord a cheilor de grup (GKA))*** [8]. Un *protocol de punere de acord a cheilor de grup (GKA)* este un protocol de stabilire a cheilor de grup în care cheia secretă comună se obţine în urma contribuţiei tuturor entităţilor autorizate astfel încât nici una să nu poată predetermina valoarea acesteia.

Schemele de partajare a secretelor se utilizează în construcţia protocoalelor de stabilire a cheilor de grup, permiţând construcţii eficiente: utilizatorii pot comunica numai prin canale de tip broadcast, calculul cheii se poate reduce la simple ecuaţii liniare, numărul de runde poate rămâne constant indiferent de numărul de participanţi. În plus, oferă o modalitate simplă de a diferenţia membrii în funcţie de puterea deţinută în cadrul grupului, permit delegarea sarcinilor, detecţia trişorilor şi o dimensionare facilă a grupului în funcţie de un prag [21].

Multiple construcţii GKE bazate pe scheme de partajare a secretelor există în literatură, unele dintre acestea fiind prezentate în varianta completă a tezei. În continuare ne vom referi strict la cele legate de contribuţiile personale.

[4]Group Key Transfer.
[5]Group Key Agreement.

Capitolul 5

Protocoale GKE informal sigure bazate pe SSS

Tabelul 5.1 sumarizează rezultatele la care se face referire pe parcursul capitolului.

Tabelul 5.1: Atacuri şi îmbunătăţiri aplicate protocoalelor GKE informal sigure

Protocolul original	Atacuri şi Îmbunătăţiri
Harn şi Lin (Jun 2010) [4]	Nam et al. (Dec 2011) [9]
	Yuan et al. (Sep 2013) [29], Olimid [17]
Hsu et al. (Jan 2012) [5]	Olimid [19]
Sun et al. (Mar 2012) [23]	Olimid (Mar 2013) [18], Kim et al. (May 2013) [7]
Yuan et al. (Jan 2013) [28]	Olimid (Jul 2013) [10], [15]

5.1 Harn şi Lin (2010)

Harn şi Lin au introdus în 2010 un protocol GKT [4] bazat pe schema de partajare a lui Shamir [22]. Nam et al. arată vulnerabilitatea protocolului la un atac prin reluare [9]. Recent, Yuan et al. introduc un atac de tip man-in-the-middle şi propun o modalitate de îmbunătăţire a protocolului pentru a rezista la atac [29]. Fig.5.1 descrie varianta Yuan et al. a protocolului lui Harn şi Lin.

Contrar a ceea ce afirmă autorii, varianta propusă rămâne vulnerabilă la un atac de tip man-in-the-middle, prezentat în detaliu în Fig.5.2 [17].

5.2 Hsu et al. (2012)

Hsu et al. au introdus în anul 2012 un protocol GKT bazat pe perspectiva non-tradiţională a schemelor de m-partajare prezentată în Secţiunea 2.1 [5]. Fig 5.3 descrie protocolul în detaliu.

Iniţializare. KGC selectează 2 numere prime sigure p şi q (i.e. $p' = \frac{p-1}{2}$ şi $q' = \frac{q-1}{2}$ sunt de asemenea prime) şi calculează $n = pq$;
Apoi, alege $e \in \mathbb{Z}_n^*$ a.î. $(e, \Phi(n)) = 1$ şi calculează $d \in \mathbb{Z}_n^*$ a.î. $ed = 1 \pmod{\Phi(n)}$;

Înregistrare. Fiecare utilizator $U_i, i = 1, \ldots, m$, stabileşte un secret de lungă durată $(x_i, y_i) \in \mathbb{Z}_n^* \times \mathbb{Z}_n^*$ cu KGC şi primeşte cheia sa publică (e, n);

Runda 1. Utilizatorul U_1:
1.1. transmite:
$U_1 \rightarrow KGC : \{U_1, \ldots, U_t\}$

Runda 2. KGC:
2.1. calculează $v = h(U_1, \ldots, U_t)^d \pmod{n}$;
2.2. transmite:
$KGC \rightarrow^*: (\{U_1, \ldots, U_t\}, v)$

Runda 3. Fiecare utilizator $U_i, i = 1, \ldots, t$:
3.1. verifică dacă $h(U_1, \ldots, U_t) = v^e \pmod{n}$;
Dacă egalitatea nu se satisface, abandonează.
3.2. alege $R_i \leftarrow^R \mathbb{Z}_n^*$;
3.3. transmite:
$U_i \rightarrow KGC : R_i$

Runda 4. KGC:
4.1. alege o cheie $k \leftarrow^R \mathbb{Z}_n^*$;
4.2. generează polinomul $f(x)$ de grad t care trece prin $t + 1$ puncte $(0, k)$, $(x_1, y_1 \oplus R_1)$, $\ldots$, $(x_t, y_t \oplus R_t)$;
4.3. calculează t puncte adiţionale $P_1, \ldots, P_t$ ale lui $f(x)$;
4.4. calculează mesajul de autentificare $Auth = h(k, U_1, \ldots, U_t, R_1, \ldots, R_t, P_1, \ldots, P_t)$;
4.5. transmite:
$KGC \rightarrow^*: (P_1, \ldots, P_t, R_1, \ldots, R_t, Auth)$

Calculul cheii. Fiecare utilizator U_i, $i = 1, \ldots, t$:
5.1. calculează cheia de grup $k = f(0)$ prin interpolarea punctelor $P_1, \ldots, P_t$ şi $(x_i, y_i \oplus R_i)$;
5.2. verifică dacă $Auth = h(k, U_1, \ldots, U_t, R_1, \ldots, R_t, P_1, \ldots, P_t)$;
Dacă egalitatea nu se satisface, abandonează.

Figura 5.1: Versiunea Yuan et al. a protocolului de transfer de chei Harn şi Lin [29]

Pas 1. U_a nu intervine în Runda 1 a unei sesiuni legitime, dar interceptează răspunsul $(\{U_1, \ldots, U_t\}, v)$ transmis în Runda 2 şi previne ca acesta să ajungă la participanţi.

Pas 2. U_a joacă rolul fiecărui particiant la sesiune şi trimite o valoare R'_j în numele lui U_j, $j = 1, \ldots, t$:

$$U_a(U_j) \rightarrow KGC : R'_j$$

Pas 3. U_a interceptează mesajul $(R'_1, \ldots, R'_t, P'_1, \ldots, P'_t, Auth')$ transmis în Runda 4 şi previne ca acesta să ajungă la utilizatori.

Pas 4. U_a iniţiază o nouă sesiune:

$$U_a \rightarrow KGC : \{U_1, \ldots, U_{i-1}, U_a, U_{i+1}, \ldots, U_t\}$$

Pas 5. U_a interceptează mesajul $(\{U_1, \ldots, U_{i-1}, U_a, U_{i+1}, \ldots, U_t\}, v')$ transmis în Runda 2 a noii sesiuni şi previne ca acesta să ajungă la utilizatori;

Pas 6. U_a transmite mesajul de la Pasul 1 tuturor membrilor, cu excepţia lui U_i; Utilizatorii U_j, $j = 1, \ldots, t$, $j \neq i$, verifică $h(U_1, \ldots, U_t) = v^e \pmod n$ (şi deci nu poate detecta atacul);

Pas 7. U_a permite mesajelor R_j, $j = 1, \ldots, t$, $j \neq i$ să ajungă la KGC, alege $R_a \leftarrow^R \mathbb{Z}_n^*$ şi transmite către KGC :

$$U_a \rightarrow KGC : R_a$$

Pas 8. U_a interceptează mesajul $(R_1, \ldots, R_{i-1}, R_a, R_{i+1}, \ldots, R_t, P_1, \ldots, P_t, Auth)$ în Runda 4, calculează cheia de grup $k = f(0)$ prin interpolarea punctelor $P_1, \ldots, P_t$ şi $(x_a, y_a \oplus R_a)$, apoi transmite mesajul participanţilor (ca provenind de la KGC).

Pas 9. Utilizatorii U_j, $j = 1, \ldots, t$, $j \neq i$, determină cheia de grup $k = f(0)$ prin interpolarea punctelor $P_1, \ldots, P_t$ şi $(x_j, y_j \oplus R_j)$, verifică dacă $Auth = h(k, U_1, \ldots, U_t, R_1, \ldots, R_{i-1}, R_a, R_{i+1}, \ldots R_t, P_1, \ldots, P_t)$ se satisface şi acceptă k drept cheie corectă de grup.

Figura 5.2: Atac man-in-the-middle asupra versiunii Yuan et al. a protocolului Harn şi Lin [17]

Protocolul Hsu et al. este vulnerabil la un atac activ din interior, prezentat în Fig.5.4 [19]. Atacul este cauzat de o slăbiciune de autentificare: cheia de grup nu este autentificată ca provenind de la iniţiatorul U_1. Acest fapt îi permite adversarului să joace rolul lui U_1 şi să trimită un mesaj modificat, dar aparent autentic. Propunem în Fig.5.5 o variantă îmbunătăţită a protocolului care rezistă atacului.

Inţializare. Fie G un grup ciclic multiplicativ de ordin p, cu g generator, unde p este un număr prim sigur mare (i.e. $p' = \frac{p-1}{2}$ este prim);

Înregistrare. Fiecare utilizator U_i, $i = 1, \ldots, m$ deţine o pereche cheie publică - cheie privată (pk_i, sk_i) a.î. $pk_i = g^{sk_i}$ în G.

Runda 1. Utilizatorul U_1:

1.1. alege $r_1 \leftarrow^R \mathbb{Z}_p^*$;

1.2. transmite o cerere de generare a cheii:

$U_1 \rightarrow^*$: $(\{U_1, \ldots, U_t\}, r_1, pk_1)$

Runda 2. Fiecare utilizator $U_i, i = 2, \ldots, t$:

2.1. alege $r_i \leftarrow^R \mathbb{Z}_p^*$;

2.2. calculează $S_i = pk_1^{sk_i r_i r_1}$ un secret comun cu U_1 şi $Auth_i = h(S_i, r_1)$;

2.3. transmite:

$U_i \rightarrow^*$: $(r_i, pk_i, Auth_i)$

Runda 3. Utilizatorul U_1:

3.1. calculează $S_i = pk_i^{sk_1 r_i r_1}$, $i = 2, \ldots, t$;

3.2. verifică dacă $Auth_i = h(S_i, r_1)$, $i = 2, \ldots, t$;
Dacă cel puţin o egalitate nu se satisface, abandonează;

3.3. alege cheia de grup $k \leftarrow^R \mathbb{Z}_p^*$, împarte fiecare secret S_i în două părţi $S_i = x_i || y_i$ şi calculează $t - 1$ valori $K_i = k - T_i$, unde $T_i = (y_i v(x_i), r)$ este produsul scalar al vectorilor $y_i v(x_i) = y_i \sum_{j=1}^{t} x_i^{j-1} e_j$ ($e_j = (0, \ldots, 1, \ldots, 0)$ cu 1 pe poziţia j), $i = 2, \ldots, t$ şi $r = (r_1, \ldots, r_t)$;

3.4. calculează $Auth = h(k, U_1, \ldots, U_t, r_1, \ldots, r_t, K_2, \ldots, K_t)$;

3.5. transmite:

$U_1 \rightarrow^*$: $(Auth, K_2, \ldots, K_t)$

Calculul Cheii. Fiecare utilizator U_i, $i = 2, \ldots, t$:

4.1. calculează produsul scalar $T_i = (y_i v(x_i), r)$, determină cheia de grup $k = T_i + K_i$;

4.2. verifică dacă $Auth = h(k, U_1, \ldots, U_t, r_1, \ldots, r_t, K_2, \ldots, K_t)$;
Dacă egalitatea nu se satisface, abandonează.

Figura 5.3: Versiunea originală a protocolului de transfer de chei Hsu et al. [5]

Pas 1. U_a este autorizat să participe la o sesiune a protocolului şi să determine cheia k:
$k = T_a + K_a$

Pas 2. U_a interceptează mesajul transmis în Runda 3 a protocolului şi îl opreşte să ajungă la U_i;

pas 3. U_a interceptează K_i şi calculează:
$T_i = k - K_i$

Pas 4. U_a alege o cheie k_i, calculează valoarea corespunzătoare $K_i' = k_i - T_i$ şi îşi autentifică alegerea ca:
$Auth_i = h(k_i, U_1, \ldots, U_t, r_1, \ldots, r_t, K_2, \ldots, K_i', \ldots K_t)$

Pas 5. U_a transmite:
$U_a \rightarrow U_i : (Auth_i, K_2, \ldots, K_i', \ldots, K_t)$

Pas 6. U_i determină valoarea corectă T_i, calculează cheia de grup $k_i = T_i + K_i'$ şi verifică dacă se satisface
$Auth = h(k_i, U_1, \ldots, U_t, r_1, \ldots, r_t, K_2, \ldots, K_i', \ldots, , K_t)$
U_i acceptă k_i ca fiind cheia comună de grup.

Figura 5.4: Atac din interior asupra versiunii originale a protocolului Hsu et al. [19]

5.3 Sun et al. (2012)

Sun et al. au introdus în anul 2012 un protocol GKT bazat pe perspectiva non-tradiţională a schemelor de m-partajare prezentată în Secţiunea 2.1 [23]. Fig.5.6 descrie protocolul în detaliu.

Protocolul Sun et al.'s este vulnerabil la un atac din interior, prezentat în Fig.5.7 şi la un atac de tip *cheie cunoscută*[1], detaliat în Fig.5.8 [18]. Ambele atacuri sunt cauzate de faptul că valorile $g^{s_i'}$, $i = 1, \ldots, m$ se menţin pentru mai multe sesiuni. Propunem în Fig.5.9 o modalitate de eliminare a acestei probleme [18] inspirată din lucrarea lui Pieprzyk şi Li [21].

5.4 Yuan et al. (2013)

Yuan et al. au introdus în anul 2013 un protocol GKT bazat pe schema de partajare Shamir [22] în care utilizatorii, ca urmare a înregistrării în grup, deţin parole de lungă durată [28]. Fig.5.10 descrie protocolul în detaliu.

Protocolul este vulnerabil la un *atac prin reluare*[2], prezentat în Fig.5.11 [10], [15]. Atacul

[1] Known key attack.
[2] Replay attack.

Iniţializare. Fie G un grup ciclic multiplicativ de ordin p, cu g generator, unde p este un număr prim sigur mare (i.e. $p' = \frac{p-1}{2}$ este prim);

Înregistrare. Fiecare utilizator U_i, $i = 1, \ldots, m$ deţine o pereche cheie publică - cheie privată (pk_i, sk_i) a.î. $pk_i = g^{sk_i}$ în G.

Runda 1. Utilizatorul U_1:

1.1. alege $r_1 \leftarrow^R \mathbb{Z}_p^*$;

1.2. transmite o cerere de generare a cheii:

$U_1 \rightarrow^*$: $(\{U_1, \ldots, U_t\}, r_1, pk_1)$

Runda 2. Fiecare utilizator $U_i, i = 2, \ldots, t$:

2.1. alege $r_i \leftarrow^R \mathbb{Z}_p^*$;

2.2. calculează $S_i = pk_1^{sk_i r_i r_1}$ un secret comun cu U_1 şi $Auth_i = h(S_i, r_1)$;

2.3. transmite:

$U_i \rightarrow^*$: $(r_i, pk_i, Auth_i)$

Runda 3. Utilizatorul U_1:

3.1. calculează $S_i = pk_i^{sk_1 r_i r_1}$, $i = 2, \ldots, t$;

3.2. verifică dacă $Auth_i = h(S_i, r_1)$, $i = 2, \ldots, t$;
Dacă cel puţin o egalitate nu se satisface, abandonează;

3.3. alege cheia de grup $k \leftarrow^R \mathbb{Z}_p^*$, împarte fiecare secret S_i în două părţi $S_i = x_i || y_i$ şi calculează $t-1$ valori $K_i = k - T_i$, unde $T_i = (y_i v(x_i), r)$ este produsul scalar al vectorilor $y_i v(x_i) = y_i \sum_{j=1}^{t} x_i^{j-1} e_j$ ($e_j = (0, \ldots, 1, \ldots, 0)$ cu 1 pe poziţia j), $i = 2, \ldots, t$ şi $r = (r_1, \ldots, r_t)$;

3.4. calculează $Auth = \Sigma.\mathsf{Sign}_{U_1}(h(k, U_1, \ldots, U_t, r_1, \ldots, r_t, K_2, \ldots, K_t))$;

3.5. transmite:

$U_1 \rightarrow^*$: $(Auth, K_2, \ldots, K_t)$

Calculul Cheii. Fiecare utilizator U_i, $i = 2, \ldots, t$:

4.1. calculează produsul scalar $T_i = (y_i v(x_i), r)$, determină cheia de grup $k = T_i + K_i$;

4.2. verifică dacă $\Sigma.\mathsf{Verify}_{U_1}(h(k, U_1, \ldots, U_t, r_1, \ldots, r_t, K_2, \ldots, K_t), Auth) = 1$.
Daca egalitatea nu se satisface, abandonează.

Figura 5.5: Versiunea îmbunătăţită a protocolului de transfer de chei Hsu et al. [19]

Iniţializare. KGC alege un grup ciclic multiplicativ G de ordin prim p cu g generator;

Înregistrare. Fiecare utilizator $U_i, i = 1, \ldots, m$, stabileşte un secret de lungă durată $s'_i \in G$ comun cu KGC;

Runda 1. Utilizatorul U_1:
1.1. transmite o cerere de generare a cheii:
$U_1 \rightarrow KGC : \{U_1, \ldots, U_t\}$

Runda 2. KGC:
2.1. transmite:
$KGC \rightarrow^*: \{U_1, \ldots, U_t\}$

Runda 3. Fiecare utilizator $U_i, i = 1, \ldots, t$:
3.1. alege $r_i \leftarrow^R \mathbb{Z}_p^*$;
3.2. transmite:
$U_i \rightarrow^*: r_i$

Runda 4. KGC:
4.1. alege $S \leftarrow^R G$;
4.2. partajează S în 2 componente de t ori a.î. $S = s_i + s'_i$, $i = 1, \ldots, t$;
4.3. calculează $k = g^S$, $M_i = (g^{s_i+r_i}, U_i, h(U_i, g^{s_i+r_i}, s'_i, r_i))$, $i = 1, \ldots, t$ şi $Auth = h(k, g^{s_1+r_1}, \ldots, g^{s_t+r_t}, U_1, \ldots, U_t, r_1, \ldots, r_t)$;
4.4. transmite:
$KGC \rightarrow^*: (M_1, \ldots, M_t, Auth)$

Calculul cheii. Fiecare utilizator U_i, $i = 1, \ldots, t$:
5.1. verifică dacă $h(U_i, g^{s_i+r_i}, s'_i, r_i)$ este egal cu valoarea corespunzătoare din M_i; Dacă egalitatea nu se satisface, abandonează;
5.2. calculează cheia de grup $k = g^{s'_i} g^{s_i+r_i} (g^{r_i})^{-1}$;
5.3. verifică dacă $Auth = h(k, g^{s_1+r_1}, \ldots, g^{s_t+r_t}, U_1, \ldots, U_t, r_1, \ldots, r_t)$; Dacă egalitatea nu se satisface, abandonează;
5.4. calculează $h_i = h(s'_i, k, U_1, \ldots, U_t, r_1, \ldots, r_t)$
5.4. transmite:
$U_i \rightarrow KGC : h_i$

Confirmarea cheii. KGC:
6.1. verifică dacă $h_i = h(s'_i, k, U_1, \ldots, U_t, r_1, \ldots, r_t)$, $i = 1, \ldots, t$, certificând că toţi utilizatorii deţin aceeaşi cheie.

Figura 5.6: Versiunea originală a protocolului de transfer de chei Sun et al. [23]

Pas 1. U_a este autorizat să participe la o sesiune (s_1) şi să determine cheia $k_{(s_1)}$:

$$k_{(s_1)} = \frac{g^{s'_a} \cdot g^{s_{a(s_1)} + r_{a(s_1)}}}{g^{r_{a(s_1)}}}$$

Pas 2. Cum $r_{i_{(s_1)}}$ şi $g^{s_{i(s_1)} + r_{i(s_1)}}$ sunt transmise în clar în Rundele 3 şi 4 ale protocolului, U_a poate calcula $g^{s'_i}$ pentru orice $U_i \in \mathcal{U}_{(s_1)}$:

$$g^{s'_i} = \frac{k_{(s_1)} \cdot g^{r_{i(s_1)}}}{g^{s_{i(s_1)} + r_{i(s_1)}}}$$

Pas 3. U_a interceptează $r_{j_{(s_2)}}$ şi $g^{s_{j(s_2)} + r_{j(s_2)}}$ într-o sesiune (s_2) a.î. $U_a \notin \mathcal{U}_{(s_2)}$ şi determină, pentru orice $U_j \in \mathcal{U}_{(s_2)}$:

$$g^{s_{j(s_2)}} = \frac{g^{s_{j(s_2)} + r_{j(s_2)}}}{g^{r_{j(s_2)}}}$$

Pas 4. Dacă există $U_b \in \mathcal{U}_{(s_1)} \bigcap \mathcal{U}_{(s_2)}$, atunci U_a calculează cheia $k_{(s_2)}$ a sesiunii (s_2) ca:

$$k_{(s_2)} = g^{s'_b} \cdot g^{s_{b(s_2)}} = g^{s'_b + s_{b(s_2)}}.$$

Figura 5.7: Atac din interior asupra versiunii originale a protocolului Sun et al. [18]

Pas 1. Cum $r_{i_{(s_1)}}$ şi $g^{s_{i(s_1)} + r_{i(s_1)}}$ sunt transmise în clar în Rundele 3 şi 4 ale protocolului, U_a poate calcula $g^{s'_i}$ pentru orice $U_i \in \mathcal{U}_{(s_1)}$ sub premisa că a reuşit prin diferite mijloace să determine cheia de sesiune $k_{(s_1)}$:

$$g^{s'_i} = \frac{k_{(s_1)} \cdot g^{r_{i(s_1)}}}{g^{s_{i(s_1)} + r_{i(s_1)}}}$$

Pas 2. U_a interceptează $r_{j_{(s_2)}}$ şi $g^{s_{j(s_2)} + r_{j(s_2)}}$ într-o sesiune (s_2), a.î. $(s_2) \neq (s_1)$, $U_a \notin \mathcal{U}_{(s_2)}$ şi determină, pentru orice $U_j \in \mathcal{U}_{(s_2)}$:

$$g^{s_{j(s_2)}} = \frac{g^{s_{j(s_2)} + r_{j(s_2)}}}{g^{r_{j(s_2)}}}$$

Pas 3. Dacă există $U_b \in \mathcal{U}_{(s_1)} \bigcap \mathcal{U}_{(s_2)}$, atunci U_a determină cheia $k_{(s_2)}$ a sesiunii (s_2) ca:

$$k_{(s_2)} = g^{s'_b} \cdot g^{s_{b(s_2)}} = g^{s'_b + s_{b(s_2)}}.$$

Figura 5.8: Atac cu cheie cunoscută asupra versiunii originale a protocolului Sun et al. [18]

este posibil pentru că KGC nu poate detecta reutilizarea mesajelor. Propunem în Fig.5.12 o modalitate de eliminare a acestei slăbiciuni [10], [15] analoagă celei introduse de Nam et al. [9] pentru protocolul similar al lui Harn şi Lin [4].

Versiunea propusă rămâne însă vulnerabilă la un atac din interior, prezentat în Fig.5.13 [15]. Acesta este cauzat de faptul că pw_{iy} poate fi exprimat ca un polinom cu coeficienţi cunoscuţi în pw_{ix}, ceea ce permite atacatorului să obţină un sistem de ecuaţii în singura

Iniţializare. KGC alege un grup ciclic multiplicativ G de ordin prim p cu g generator;

Înregistrare. Fiecare utilizator $U_i, i = 1, \ldots, m$, stabileşte un secret de lungă durată $s'_i \in G$ comun cu KGC;

Runda 1. Utilizatorul U_1:
1.1. transmite o cerere de generare a cheii:
$U_1 \rightarrow KGC : \{U_1, \ldots, U_t\}$

Runda 2. KGC:
2.1. transmite:
$KGC \rightarrow^*: \{U_1, \ldots, U_t\}$

Runda 3. Fiecare utilizator $U_i, i = 1, \ldots, t$:
3.1. alege $r_i \leftarrow^R \mathbb{Z}_p^*$;
3.2. transmite:
$U_i \rightarrow^*: r_i$

Runda 4. KGC:
4.1. alege $S \leftarrow^R G$ şi $\alpha \leftarrow^R G$;
4.2. partajează S în 2 componente de t ori a.î. $S = s_i + s'_i$, $i = 1, \ldots, t$;
4.3. calculează $k = \alpha^S$, $M_i = (\alpha^{s_i+r_i}, U_i, h(U_i, \alpha^{s_i+r_i}, s'_i, r_i, \alpha))$, $i = 1, \ldots, t$ şi $Auth = h(k, \alpha^{s_1+r_1}, \ldots, \alpha^{s_t+r_t}, U_1, \ldots, U_t, r_1, \ldots, r_t, \alpha)$;
4.4. transmite:
$KGC \rightarrow^*: (M_1, \ldots, M_t, Auth, \alpha)$

Calculul cheii. Fiecare utilizator U_i, $i = 1, \ldots, t$:
5.1. verifică dacă $h(U_i, g^{s_i+r_i}, s'_i, r_i, \alpha)$ este egal cu valoarea corespunzătoare din M_i; Dacă egalitatea nu se satisface, abandonează;
5.2. calculează cheia de grup $k = \alpha^{s'_i}\alpha^{s_i+r_i}(\alpha^{r_i})^{-1}$;
5.3. verifică dacă $Auth = h(k, \alpha^{s_1+r_1}, \ldots, \alpha^{s_t+r_t}, U_1, \ldots, U_t, r_1, \ldots, r_t, \alpha)$; Dacă egalitatea nu se satisface, abandonează;
5.4. calculează $h_i = h(s'_i, k, U_1, \ldots, U_t, r_1, \ldots, r_t, \alpha)$
5.4. transmite:
$U_i \rightarrow KGC : h_i$

Confirmarea cheii. KGC:
6.1. verifică dacă $h_i = h(s'_i, k, U_1, \ldots, U_t, r_1, \ldots, r_t, \alpha)$, $i = 1, \ldots, t$, certificând că toţi utilizatorii deţin aceeaşi cheie.

Figura 5.9: Versiunea îmbunătăţită a protocolului de transfer de chei Sun et al. [18]

Iniţializare. KGC alege 2 numere prime mari p şi q şi calculează $n = pq$;

Înregistrarea. Fiecare utilizator $U_i, i = 1, \ldots, m$, stabileşte o parolă de lungă durată $pw_i = pw_{ix}||pw_{iy}$ cu KGC;

Runda 1. Utilizatorul U_1:
1.1. alege $k_1 \leftarrow^R \mathbb{Z}_n$;
1.2. calculează $K_1 = pw_{1x} + k_1$ şi $M_1 = h_1(U_1, \ldots, U_t, k_1)$;
1.3. transmite o cerere de generare a cheii:
$U_1 \rightarrow KGC : (U_1, \{U_1, \ldots, U_t\}, K_1, M_1)$

Runda 2. KGC:
2.1. calculează $k_1 = K_1 - pw_{1x}$;
2.2. verifică dacă $M_1 = h_1(U_1, \ldots, U_t, k_1)$;
Dacă egalitatea nu se satisface, abandonează;
2.3. transmite:
$KGC \rightarrow^*$: $\{U_1, \ldots, U_t\}$

Runda 3. Fiecare utilizator $U_i, i = 2, \ldots, t$:
3.1. alege $k_i \leftarrow^R \mathbb{Z}_n$;
3.2. calculează $K_i = pw_{ix} + k_i$ şi $M_i = h_1(U_1, \ldots, U_t, k_i)$;
3.3. transmite:
$U_i \rightarrow KGC : (U_i, \{U_1, \ldots, U_t\}, K_i, M_i)$

Runda 4. KGC:
4.1. calculează $k_i = K_i - pw_{ix}$, i=2,. . . ,t;
4.2. verifică dacă $M_i = h_1(U_1, \ldots, U_t, k_i)$, i=2,. . . ,t;
Dacă egalitatea nu se satisface, abandonează;
4.3. alege 2 numere aleatoare x_{ta} şi y_{ta} de lungime egală cu pw_{ix} şi pw_{iy};
4.4. generează polinomul $f(x)$ de grad t care trece prin $t + 1$ puncte (x_{ta}, y_{ta}), $(pw_{1x}, pw_{1y} + k_1)$, . . . , $(pw_{tx}, pw_{ty} + k_t)$;
4.5. calculează t puncte adiţionale $P_1, \ldots, P_t$ ale lui $f(x)$;
4.6. calculează mesajul de verificare $V_i = h_2(U_1, \ldots, U_t, P_1, \ldots, P_t, k_i)$, $i = 1, \ldots, t$;
4.7. transmite, $i = 1, \ldots, t$:
$KGC \rightarrow U_i : (P_1, \ldots, P_t, V_i)$

Calculul cheii. Fiecare utilizator U_i, $i = 1, \ldots, t$:
5.1. verifică dacă $V_i = h_2(U_1, \ldots, U_t, P_1, \ldots, P_t, k_i)$;
Dacă egalitatea nu se satisface, abandonează;
5.2. calculează cheia de grup $k = f(0)$ prin interpolarea punctelor $P_1, \ldots, P_t$ şi $(pw_{ix}, pw_{iy} + k_i)$.

Figura 5.10: Versiunea originală a protocolului de transfer de chei Yuan et al. [28]

Pas 1. U_a iniţializează o sesiune (s_1) cerând stabilirea unei chei comune cu U_i;

Pas 2. U_a interceptează $(U_i, \{U_i, U_a\}, K_i, M_i)$ în Runda 3 a protocolului;

Pas 3. U_a iniţializează o altă sesiune (s_2) cerând stabilirea unei chei comune cu U_i şi utilizează aceeaşi valoare k_a pentru ambele sesiuni (s_1) şi (s_2);

Pas 4. U_a joacă rolul lui U_i în sesiunea (s_2) şi trimite în Runda 3 mesajul $(U_i, \{U_i, U_a\}, K_i, M_i)$ pe care l-a interceptat în pasul 2;

Pas 5. U_a este un utilizator autorizat pentru ambele sesiuni, aşa încât determină
$$f(x)_{(s_j)} = a_{(s_j)}x^2 + b_{(s_j)}x + c_{(s_j)}, j = 1, 2$$

Pas 6. Cum $(pw_{ax}, pw_{ay} + k_a)$ şi $(pw_{ix}, pw_{iy} + k_i)$ sunt puncte valide ale lui $f(x)_{(s_j)}$, U_a ştie că $f(pw_{ax})_{(s_1)} = f(pw_{ax})_{(s_2)} = pw_{ay} + k_a$ şi $f(pw_{ix})_{(s_1)} = f(pw_{ix})_{(s_2)} = pw_{iy} + k_i$;
deci pw_{ax} şi pw_{ix} sunt rădăcini pentru:
$$(a_{(s_1)} - a_{(s_2)})x^2 + (b_{(s_1)} - b_{(s_2)})x + (c_{(s_1)} - c_{(s_2)}) = 0$$

Pas 7. U_a determină parola de lungă durată a lui U_i ca:
$$pw_{ix} = pw_{ax}^{-1}(a_{(s_1)} - a_{(s_2)})^{-1}(c_{(s_1)} - c_{(s_2)}).$$
$$pw_{iy} = f(pw_{ix})_{(s_j)} - K_{i(s_j)} + pw_{ix}, \text{ for any } j = 1, 2$$

Figura 5.11: Atac prin reluare asupra versiunii originale a protocolului Yuan et al. [10]

necunoscută pw_{ix}. Propunem în Fig.5.14 o variantă îmbunătăţită a protocolului care rezistă acestui atac [10].

Iniţializare. KGC alege 2 numere prime mari p şi q şi calculează $n = pq$;

Înregistrarea. Fiecare utilizator $U_i, i = 1, \ldots, m$, stabileşte o parolă de lungă durată $pw_i = pw_{ix}||pw_{iy}$ cu KGC;

Runda 1. Utilizatorul U_1:

1.1. transmite o cerere de generare a cheii:

$$U_1 \rightarrow KGC : \{U_1, \ldots, U_t\}$$

Runda 2. KGC:

2.1. alege $k_0 \leftarrow^R \mathbb{Z}_n$;

2.2. transmite:

$$KGC \rightarrow^* : (\{U_1, \ldots, U_t\}, k_0)$$

Runda 3. Fiecare utilizator $U_i, i = 1, \ldots, t$:

3.1. alege $k_i \leftarrow^R \mathbb{Z}_n$;

3.2. calculează $K_i = pw_{ix} + k_i$ şi $M_i = h_1(U_1, \ldots, U_t, k_i, k_0)$;

3.3. transmite:

$$U_i \rightarrow KGC : (U_i, \{U_1, \ldots, U_t\}, K_i, M_i)$$

Runda 4. KGC:

4.1. calculează $k_i = K_i - pw_{ix}$, i=1,. . . ,t;

4.2. verifică dacă $M_i = h_1(U_1, \ldots, U_t, k_i, k_0)$, i=1,. . . ,t;
Dacă egalitatea nu se satisface, abandonează;

4.3. alege 2 numere aleatoare x_{ta} şi y_{ta} de lungime egală cu pw_{ix} şi pw_{iy};

4.4. generează polinomul $f(x)$ de grad t care trece prin $t + 1$ puncte (x_{ta}, y_{ta}), $(pw_{1x}, pw_{1y} + k_1)$, ... , $(pw_{tx}, pw_{ty} + k_t)$;

4.5. calculează t puncte adiţionale $P_1, \ldots, P_t$ ale lui $f(x)$;

4.6. calculează mesajul de verificare $V_i = h_2(U_1, \ldots, U_t, P_1, \ldots, P_t, k_i, k_0)$, $i = 1, \ldots, t$;

4.7. transmite, $i = 1, \ldots, t$:

$$KGC \rightarrow U_i : (P_1, \ldots, P_t, V_i)$$

Calculul Cheii. Fiecare utilizator U_i, $i = 1, \ldots, t$:

5.1. verifică dacă $V_i = h_2(U_1, \ldots, U_t, P_1, \ldots, P_t, k_i, k_0)$;
Dacă egalitatea nu se satisface, abandonează;

5.2. calculează cheia de grup $k = f(0)$ prin interpolarea punctelor $P_1, \ldots, P_t$ şi $(pw_{ix}, pw_{iy} + k_i)$.

Figura 5.12: Prima versiune îmbunătăţită a protocolului Yuan et al.'s Protocol [10]

Pas 1. U_a iniţializează 4 sesiuni (s_j), $j = 1, \ldots, 4$, cerând stabilirea unei chei comune cu U_i;

Pas 2. U_a este un utilizator autorizat pentru toate cele 4 sesiuni, aşa încât determină:
$$f(x)_{(s_j)} = a_{(s_j)}x^2 + b_{(s_j)}x + c_{(s_j)}, j = 1, \ldots, 4$$

Pas 3. Cum $(pw_{ix}, pw_{iy} + k_{i(s_j)})$ sunt puncte valide ale lui $f(x)_{(s_j)}$, U_a obţine:
$$pw_{iy} + k_{i(s_j)} = a_{(s_j)}pw_{ix}^2 + b_{(s_j)}pw_{ix} + c_{(s_j)},\ j = 1, \ldots, 4$$

Pas 4. U_a interceptează $K_{i(s_j)}$, ştie că $k_{i(s_j)} = K_{i(s_j)} - pw_{ix}$ şi determină:
$$pw_{iy} = a_{(s_j)}pw_{ix}^2 + (b_{(s_j)} + 1)pw_{ix} + c_{(s_j)} - K_{i(s_j)}, j = 1, \ldots, 4$$

Pas 5. U_a elimină pw_{iy} din primele 2 ecuaţii $(j = 1, 2)$, respectiv din ultimele 2 $(j = 3, 4)$ şi obţine:
$$A_{(s_1s_2)}pw_{ix}^2 + B_{(s_1s_2)}pw_{ix} + C_{(s_1s_2)} = 0$$
$$A_{(s_3s_4)}pw_{ix}^2 + B_{(s_3s_4)}pw_{ix} + C_{(s_3s_4)} = 0$$
unde:

$A_{(s_1s_2)} = a_{(s_1)} - a_{(s_2)}$	$A_{(s_3s_4)} = a_{(s_3)} - a_{(s_4)}$
$B_{(s_1s_2)} = b_{(s_1)} - b_{(s_2)}$	$B_{(s_3s_4)} = b_{(s_3)} - b_{(s_4)}$
$C_{(s_1s_2)} = c_{(s_1)} - c_{(s_2)} - (K_{i(s_1)} - K_{i(s_2)})$	$C_{(s_3s_4)} = c_{(s_3)} - c_{(s_4)} - (K_{i(s_3)} - K_{i(s_4)})$

Pas 6. U_a determină parola de lungă durată a lui U_i ca:
$$pw_{ix} = (A_{(s_1s_2)}C_{(s_3s_4)} - A_{(s_3s_4)}C_{(s_1s_2)})(A_{(s_3s_4)}B_{(s_1s_2)} - A_{(s_1s_2)}B_{(s_3s_4)})^{-1}$$
$$pw_{iy} = f(pw_{ix})_{(s_j)} - K_{i(s_j)} + pw_{ix}, \text{ for any } j = 1, \ldots, 4$$

Figura 5.13: Atac din interior asupra primei versiuni îmbunătăţite a protocolului Yuan et al. [15]

Iniţializare. KGC alege 2 numere prime mari p şi q şi calculează $n = pq$;

Înregistrare. Fiecare utilizator $U_i, i = 1, \ldots, m$, stabileşte o parolă de lungă durată $pw_i = pw_{ix}||pw_{iy}$ cu KGC;

Runda 1. Utilizatorul U_1:

1.1. transmite o cerere de generare a cheii:

$$U_1 \rightarrow KGC : \{U_1, \ldots, U_t\}$$

Runda 2. KGC:

2.1. alege $k_0 \leftarrow^R \mathbb{Z}_n$;

2.2. transmite:

$$KGC \rightarrow^* : (\{U_1, \ldots, U_t\}, k_0)$$

Runda 3. Fiecare utilizator $U_i, i = 1, \ldots, t$:

3.1. alege $k_i \leftarrow^R \mathbb{Z}_n$;

3.2. calculează $K_i = pw_{ix} + k_i$ şi $M_i = h_1(U_1, \ldots, U_t, k_i, k_0)$;

3.3. transmite:

$$U_i \rightarrow KGC : (U_i, \{U_1, \ldots, U_t\}, K_i, M_i)$$

Runda 4. KGC:

4.1. calculează $k_i = K_i - pw_{ix}$, i=1,. . . ,t;

4.2. verifică dacă $M_i = h_1(U_1, \ldots, U_t, k_i, k_0)$, i=1,. . . ,t;
Dacă egalitatea nu se satisface, abandonează;

4.3. alege 2 numere aleatoare x_{ta} şi y_{ta} de lungime egală cu pw_{ix} şi pw_{iy};

4.4. generează polinomul $f(x)$ de grad t care trece prin $t + 1$ puncte (x_{ta}, y_{ta}), $(pw_{1x}, h_3(U_1, \ldots, U_t, pw_{1y}, k_1, k_0))$, . . . , $(pw_{tx}, h_3(U_1, \ldots, U_t, pw_{ty}, k_t, k_0))$;

4.5. calculează t puncte adiţionale $P_1, \ldots, P_t$ ale lui $f(x)$;

4.6. calculează mesajul de verificare $V_i = h_2(U_1, \ldots, U_t, P_1, \ldots, P_t, k_i, k_0)$, $i = 1, \ldots, t$;

4.7. transmite, $i = 1, \ldots, t$:

$$KGC \rightarrow U_i : (P_1, \ldots, P_t, V_i)$$

Calculul cheii. Fiecare utilizator U_i, $i = 1, \ldots, t$:

5.1. verifică dacă $V_i = h_2(U_1, \ldots, U_t, P_1, \ldots, P_t, k_i, k_0)$;
Dacă egalitatea nu se satisface, abandonează;

5.2. calculează cheia de grup $k = f(0)$ prin interpolarea punctelor $P_1, \ldots, P_t$ şi $(pw_{ix}, h_3(U_1, \ldots, U_t, pw_{iy}, k_i, k_0))$.

Figura 5.14: A doua versiune îmbunătăţită a protocolului Yuan et al. [10]

Capitolul 6

Protocoale GKE formal sigure bazate pe SSS

6.1 Un nou protocol

Propunem un nou protocol GKA bazat pe scheme de m-partajare perfecte, descris în detaliu în Fig.6.1 [20].

6.2 Demonstraţii de securitate

Protocolul propus este demonstrat sigur în cadrul modelului formal de securitate GBG. Detalii cu privire la semnificaţia noţiunilor şi demonstraţia teoremelor se regăsesc în varianta completă a lucrării.

Teorema 6.1. ***(Securitatea AKE[1])*** [20]. Dacă schema de semnătură electronică Σ este UF-CMA[2], funcţia cu trapă secretă $\mathcal{F}$ este sigură, funcţia hash H este un oracol aleator şi schema de m-partajare $\mathcal{SS}$ este perfectă, atunci protocolul propus este AKE sigur şi

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{AKE}} \leq 2m^2\mathsf{Adv}_{\mathcal{A},\Sigma}^{\mathsf{UF-CMA}} + \frac{(q_s+q_r)^2}{2^{\mathcal{K}-1}} + \frac{(m+1){q_s}^2}{2^{\mathcal{K}-1}} + 2\mathsf{Adv}_{\mathcal{A},\mathcal{F}}.$$

Teorema 6.2. ***(MA Security[3])*** [20]. Dacă schema de semnătură electronică Σ este UF-CMA şi funcţia hash H este un oracol aleator, atunci protocolul propus este MA sigur şi

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{MA}} \leq m^2\mathsf{Adv}_{\mathcal{A},\Sigma}^{\mathsf{UF-CMA}} + \frac{(q_s+q_r)^2}{2^{\mathcal{K}}} + \frac{(m+1){q_s}^2}{2^{\mathcal{K}}}.$$

[1] Authenticated Key Exchange.
[2] UnForgeable under Chosen Message Attack.
[3] Mutual Authentication.

Iniţializare. Fie $\mathcal{F} = (\mathsf{Gen}, \mathsf{F}, \mathsf{F}^{-1})$ o funcţie cu trapă secretă şi SS o schemă de m-partajare a secretelor.

Înregistrarea. Fie $\Sigma.\mathsf{Sig}_{U_i}$ algoritmul de semnătură utilizat de U_i şi $\Sigma.\mathsf{Verify}_{U_i}$ algoritmul corespunzător de verificare, $i = 1, \ldots, m$.

Runda 1. Utilizatorul U_1:
1.1. execută $\mathcal{F}.\mathsf{Gen}$ pentru a obţine o pereche cheie publică - cheie privată (pk, sk);
1.2. alege $r_1 \leftarrow^R \{0,1\}^{\mathcal{K}}$;
1.3. transmite:
$$U_1 \rightarrow^*: (\mathcal{U}, pk, \mathsf{F}(pk, r_1), \sigma = \Sigma.\mathsf{Sign}_{U_1}(\mathcal{U}||pk||\mathsf{F}(pk, r_1)));$$

Runda 2. Fiecare utilizator U_i, $i = 2, \ldots, m$:
2.1. verifică dacă $\Sigma.\mathsf{Verify}_{U_1}(\mathcal{U}||pk||\mathsf{F}(pk, r_1), \sigma) = 1$.
Dacă egalitatea nu se satisface, abandonează;
2.2. alege $r_i \leftarrow^R \{0,1\}^{\mathcal{K}}$;
2.3. transmite:
$$U_i \rightarrow^*: (\mathsf{F}(pk, r_i),\ \sigma_i = \Sigma.\mathsf{Sign}_{U_i}(\mathcal{U}||pk||\mathsf{F}(pk, r_i)));$$

Runda 3. Utilizatorul U_1:
3.1. verifică dacă $\Sigma.\mathsf{Verify}_{U_i}(\mathcal{U}||pk||\mathsf{F}(pk, r_i), \sigma_i) = 1$, $i = 2, \ldots, m$.
Dacă cel puţin o egalitate nu se satisface, abandonează;
3.2. calculează $\mathsf{sid}_{U_1} = \mathsf{F}(pk, r_1)|| \ldots ||\mathsf{F}(pk, r_m)$, $r_i = \mathsf{F}^{-1}(sk, \mathsf{F}(pk, r_i))$, cheia de sesiune $k = \mathsf{H}(\mathsf{sid}_{U_1}||r_1||r_2|| \ldots ||r_m)$ şi r_i' a.î. $\mathcal{SS}.\mathsf{Share}(k, 2) = \{r_i, r_i'\}$, $i = 2, \ldots, m$;
3.3. transmite:
$$U_1 \rightarrow^*: (r_2', \ldots r_m', \sigma' = \Sigma.\mathsf{Sign}_{U_1}(\mathcal{U}||pk||r_2'|| \ldots ||r_m'||\mathsf{sid}_{U_1}));$$

Calculul cheii. Fiecare utilizator U_i, $i = 2, \ldots, m$:
4.1. verifică dacă $\Sigma.\mathsf{Verify}_{U_j}(\mathcal{U}||pk||\mathsf{F}(pk, r_j), \sigma_j) = 1$, $j = 2, \ldots m$, $j \neq i$.
Dacă cel puţin o egalitate nu se satisface, abandonează;
4.2. calculează $\mathsf{sid}_{U_i} = \mathsf{F}(pk, r_1)|| \ldots ||\mathsf{F}(pk, r_m)$ şi $k = \mathcal{SS}.\mathsf{Rec}(r_i, r_i')$;
4.3. verifică dacă $\Sigma.\mathsf{Verify}_{U_1}(\mathcal{U}||pk||r_2'|| \ldots ||r_m'||\mathsf{sid}_{U_i}, \sigma') = 1$.
Dacă se satisface egalitatea, acceptă cheia k; altfel, abandonează;

Confirmarea cheii. Fiecare utilizator U_i, $i = 2, \ldots, m$:
5.1. calculează r_j a.î. $\mathcal{SS}.\mathsf{Share}(k, 2) = \{r_j, r_j'\}$, $j = 2, \ldots m$, $j \neq i$;
5.2. verifică dacă $\mathsf{F}(pk, r_j)$ este egal cu valoare transmisă în pasul 2.3.
Dacă cel puţin o egalitate nu se satisface, abandonează;

Figura 6.1: Protocolul de agreare a cheilor de grup Olimid [20]

Tabelul 6.1: Analiza complexităţii protocolului propus [20]

	Stocare	Calculabilitate	Transmisie
U_1	$l_{sk} + l_{sk_1} + \mathcal{K}$	$c_{\mathcal{F}.\mathsf{Gen}} + c_{\mathsf{F}} + (m-1)c_{\mathsf{F}^{-1}} + c_{\mathsf{H}} +$ $2c_{\Sigma.\mathsf{Sign}} + (m-1)c_{\Sigma.\mathsf{Verify}} + (m-1)c_{\mathcal{SS}.\mathsf{Share}}$	$l_{\mathcal{U}} + l_{pk} + ml_{\mathsf{F}} +$ $+(m+1)l_{\Sigma.\mathsf{Sign}} +$ $(m-1)\mathcal{K}$
U_i $(i \neq 1)$	$l_{sk_i} + \mathcal{K}$	$c_{\mathsf{F}} + c_{\Sigma.\mathsf{Sign}} + mc_{\Sigma.\mathsf{Verify}} + c_{\mathcal{SS}.\mathsf{Rec}} +$ $(+(m-2)c_{\mathcal{SS}.\mathsf{Share}})$	

Tabelul 6.2: Comparaţia protocolului propus cu protocoale existente [20]

	Nr. de runde	Tip de transmisiune	Grup static/dinamic	Generare sid	Model de securitate
Protocolul propus	3	broadcast	static	la rulare	GBG/ROM
Bresson-Catalano [2]	3	unicast, broadcast	static	anterior	BCP/ROM
Cao et al. [3]	3	broadcast	static	anterior	UC/ROM

Teorema 6.3. ***(Contributivitate)*** [20]. Dacă funcţia cu trapă secretă $\mathcal{F}$ este sigură şi funcţia hash H este un oracol aleator, atunci protocolul propus satisface proprietatea de contributivitate şi

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{Con}} \leq \frac{(m+1){q_s}^2}{2^{\mathcal{K}}} + \frac{q_r}{2^{\mathcal{K}}}.$$

6.3 Analiză

Tabelul 6.1 analizează complexitatea protocolului propus din trei perspective distincte: capacitate necesară de stocare, cost computaţional şi cost total de transmisie. Am notat cu l_x lungimea (în biţi) a lui x şi cu c_y costul de execuţie a lui y.

Tabelul 6.2 compară protocolul propus cu două protocoale GKA existente, care beneficiază de asemenea de o demonstraţie riguroasă de securitate, însă în modele distincte.

Concluzii şi direcţii viitoare de cercetare

Teza de faţă se axează asupra protocoalelor de stabilire a cheilor de grup bazate pe partajarea secretelor şi schemelor de partajare utilizate pentru construcţia acestora. Menţionăm pe scurt rezultatele personale şi indicăm posibile direcţii viitoare de cercetare. Precizăm că toate capitolele, cu excepţia celor introductive (Partea I, Capitolul 1 şi Partea II, Capitolul 4), conţin rezultate proprii. Împărţim rezultatele obţinute în două direcţii de studiu:

- **Criptanaliză (Capitolele 3 şi 5)**.

  Introducem un atac SETUP asupra schemelor de partajare a secretelor care utilizează destule valori aleatoare, cu rolul de a oferi unui atacator un avantaj copleşitor în determinarea secretului [14]. Deşi implementări maliţioase au fost implementate pentru multiple primitive criptografice (protocoale de generare a cheilor, semnături electronice, sisteme de vot electronic, etc.), suntem primii care le considerăm în cazul partajării de secrete. Tehnica propusă poate fi aplicată unor scheme clasice, precum Shamir, care este utilizată pentru construcţia a numeroase protocoale de stabilire a cheilor de grup. În consecinţă, analiza securităţii acestor protocoale cu privire la atacul de tip SETUP asupra schemelor pe care se bazează reprezintă un subiect imediat de cercetare.

  Motivaţi de multitudinea protocoalelor de stabilire a cheilor de grup bazate pe scheme de partajare a secretelor definite în ultimii ani şi care nu beneficiază de o demonstaţie riguroasă de securitate, propunem o analiză a acestora [10], [15], [16], [17], [18], [19]. Cum nu există argumente riguroase care să le ateste securitatea, vulnerabilităţile tind să apară în mod natural. Introducem şase astfel de atacuri asupra a patru protocoale foarte recente (sau a variantelor lor îmbunătăţite) (2012-2013) şi propunem modalităţi de remediere a acestora. Versiunilor îmbunătăţite pentru a rezista la aceste atacuri le lipseşte de asemenea o demonstraţie de securitate, ceea ce înseamnă că pot fi de asemenea uşor vulnerabile. Cercetări viitoare pot extinde succesiunile de atacuri şi contramăsuri aplicate protocoalelor analizate sau pot dezvălui atacuri similare asupra altor protocoale.

- **Construcţii (Capitolele 2 şi 6)**.

  Definim o schemă de partajare vizuală care permite împărţirea unei imagini alb-negru secrete în componente color [11], [12]. Propunerea nu oferă nici o informaţie despre secret pentru cel mult doi participanţi şi permite reconstrucţia perfectă când toţi

participanţii colaborează. O construcţie a unui protocol de stabilire a cheilor de grup bazat pe scheme vizuale de partajare poate reprezenta o abordare interesantă pentru cercetări viitoare. Atât cât cunoaştem, nu există o astfel de construcţie în literatură. Suntem conştienţi de limitările practice ale unui astfel de propuneri, dar putem indica o utilizare imediată: partajarea unei chei vizuale secrete comune care poate fi ulterior utilizată în cadrul criptării vizuale.

Prezentăm noţiunea de *scheme de m-partajare* şi introducem conceptul de *scheme de m-partajare perfecte* ca o generalizare a schemelor de partajare perfecte [20]. Cele foarte puţine protocoale de stabilire a cheilor de grup bazate pe scheme de partajare care deţin o analiză riguroasă a securităţii reprezintă o motivaţie: folosim schemele de m-partajare (datorită avantajelor pe care le introduc) ca primitivă pentru construcţia unui protocol GKA [20]. Protocolul este demonstrat sigur în cadrul modelului GBG şi menţine acelaşi număr de runde de comunicaţie ca şi protocoalele existente. În limita cunoştinţelor noastre, nici un alt protocol bazat pe scheme de partajare nu este demonstrat sigur într-un model de securitate similar. Considerăm ca direcţii viitoare de cercetare îmbunătăţirea protocolului pentru a rezista la atacuri EKL[4] (o proprietate care nu este modelată în GBG), a admite grupuri dinamice, anonimitate, robusteţe sau eficienţă crescută. Definirea unor noi protocoale sigure de stabilire a cheilor de grup care utilizează partajarea secretelor reprezintă o sarcină destul de dificilă.

Alte posibile direcţii de cercetare includ:

- *definirea protocoalelor GKE flexibile bazate pe scheme de partajare a secretelor* Protocoalele flexibile GKE au fost introduse în 2010 de către Abdalla et al. [1]: în timp ce un grup de utilizatori stabileşte o cheie secretă comună, fiecare membru obţine informaţie suplimentară pe care o poate utiliza ulterior pentru a dobândi, la cerere şi fără a iniţia o nouă sesiune, chei independente partajate de submulţimi ale mulţimii iniţiale de participanţi.
- *definirea protocoalelor GKE asimetrice (ASGKE) bazate pe scheme de partajare a secretelor.* Protocoalele asimetrice GKE au fost introduse în 2009 de către Wu et al. [25]: un grup de utilizatori partajează o cheie publică comună, dar fiecare membru deţine o cheie privată distinctă (care îi permite de exemplu să decripteze mesajele criptate folosind cheia publică comună sau să genereze o semnătură verificabilă cu cheia publică comună). Aceste sisteme prezintă proprietăţi remarcabile: monitorizare, delegare a atribuţiunilor, păstrarea cheilor de rezervă, generarea semnăturilor de grup. Menţionez că deţin un rezultat personal în acest domeniu, însă construcţia nu se bazează pe scheme de partajare a secretelor, drept urmare nu face parte direct din aria de interes a tezei de doctorat [13].
- *analizarea şi îmbunătăţirea protocoalelor GKE existente bazate pe primitive şi concepte moderne.* Lucrarea de faţă se rezumă la protocoale GKE bazate pe scheme de partajare clasice. O direcţie viitoare de studiu o constituie domeniul protocoalelor care utilizează scheme de partajare bazate pe latici, perechi biliniare sau chiar cuantice.

[4]Ephemeral Key Leakage.

Bibliografie

[1] Michel Abdalla, Céline Chevalier, Mark Manulis, and David Pointcheval. Flexible group key exchange with on-demand computation of subgroup keys. In *Proceedings of the Third international conference on Cryptology in Africa*, AFRICACRYPT'10, pages 351–368, Berlin, Heidelberg, 2010. Springer-Verlag. pages 48

[2] Emmanuel Bresson and Dario Catalano. Constant round authenticated group key agreement via distributed computation. In *Proceedings of the 7th International Workshop on Theory and Practice in Public Key Cryptography*, PKC'2004, pages 115–129. Springer, 2004. pages 45

[3] Chunjie Cao, Chao Yang, Jianfeng Ma, and Sangjae Moon. Constructing UC secure and constant-round group key exchange protocols via secret sharing. *EURASIP J. Wirel. Commun. Netw.*, 2008:4:1–4:9, January 2008. pages 45

[4] Lien Harn and Changlu Lin. Authenticated group key transfer protocol based on secret sharing. *IEEE Trans. Comput.*, 59(6):842–846, June 2010. pages 29, 36

[5] Chingfang Hsu, Bing Zeng, Qi Cheng, and Guohua Cui. A novel group key transfer protocol. Cryptology ePrint Archive, Report 2012/043, 2012. pages 29, 32

[6] Ehud D. Karnin, Jonathan W. Greene, and Martin E. Hellman. On secret sharing systems. *IEEE Transactions on Information Theory*, 29:35–41, 1983. pages 16

[7] Mijin Kim, Namje Park, and Dongho Won. Cryptanalysis of an authenticated group key transfer protocol based on secret sharing. In *Grid and Pervasive Computing*, pages 761–766. Springer-Verlag, 2013. pages 29

[8] Alfred J. Menezes, Scott A. Vanstone, and Paul C. Van Oorschot. *Handbook of Applied Cryptography*. CRC Press, Inc., 1996. pages 13, 14, 27, 28

[9] Junghyun Nam, Moonseong Kim, Juryon Paik, Woongryul Jeon, Byunghee Lee, and Dongho Won. Cryptanalysis of a group key transfer protocol based on secret sharing. In *Proceedings of the Third international conference on Future Generation Information Technology*, FGIT'11, pages 309–315, Berlin, Heidelberg, 2011. Springer-Verlag. pages 29, 36

[10] Ruxandra F. Olimid. A chain of attacks and countermeasures applied to a group key transfer protocol (abstract). In *(Pre)Proceedings of Western European Workshop on Research in Cryptology*, WEWoRC'13. pages 29, 33, 36, 39, 40, 42, 47

[11] Ruxandra F. Olimid. About a visual secret sharing scheme. In *Proceedings of the 3rd International Conference on Security for Information Technology and Communications*, SECITC'10, pages 13–17, 2010. pages 16, 17, 19, 47

[12] Ruxandra F. Olimid. Python implementation of visual secret sharing schemes. *Journal of Information Systems & Operations Management*, 5(2.1):538–550, December 2011. pages 17, 47

[13] Ruxandra F. Olimid. A new public key cryptosystem with key escrow capabilities. In *Proceedings of the 7th South East European Doctoral Student Conference*, DSC 2012, pages 610–624, 2012. pages 48

[14] Ruxandra F. Olimid. SETUP in secret sharing schemes using random number generators. 2012. Under review. pages 22, 23, 24, 47

[15] Ruxandra F. Olimid. Cryptanalysis of a password-based group key exchange protocol using secret sharing. *Appl. Math. Inf. Sci*, 7(4):1585–1590, July 2013. pages 29, 33, 36, 41, 47

[16] Ruxandra F. Olimid. On the (in)security of group key transfer protocols. In *Proceedings of the Romanian Academy, series A*, volume 14, Special Issue RCD'13, pages 378–387, 2013. pages 47

[17] Ruxandra F. Olimid. On the (non)improvement of an authenticated group key transfer protocol based on secret sharing. 2013. Under review. pages 29, 31, 47

[18] Ruxandra F. Olimid. On the security of an authenticated group key transfer protocol based on secret sharing. In *Proceedings of the 2013 international conference on Information and Communication Technology*, ICT-EurAsia'13, pages 399–408, Berlin, Heidelberg, 2013. Springer-Verlag. pages 29, 33, 36, 37, 47

[19] Ruxandra F. Olimid. On the vulnerability of a group key transfer protocol based on secret sharing. 2013. Under review. pages 29, 31, 33, 34, 47

[20] Ruxandra F. Olimid. Provable secure constant-round group key agreement protocol based on secret sharing. In *Proceedings of International Joint Conference SOCO'13 - CISIS'13 - ICEUTE'13*, AISC 239, pages 489–498. Springer, 2013. pages 15, 43, 44, 45, 48

[21] Joseph Pieprzyk and Chih-Hung Li. Multiparty key agreement protocols. *IEE Proceedings-Computers and Digital Techniques*, 147(4):229–236, 2000. pages 28, 33

[22] Adi Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, November 1979. pages 29, 33

[23] Yi Sun, Qiaoyan Wen, Hongxiang Sun, Wenmin Li, Zhengping Jin, and Hua Zhang. An authenticated group key transfer protocol based on secret sharing. *Procedia Engineering*, 29(0):403 – 408, 2012. 2012 International Workshop on Information and Electronics Engineering. pages 16, 29, 33, 35

[24] Web-site. Python programming language, Official website. Last accessed: July 2013. http://www.python.org/. pages 18

[25] Qianhong Wu, Yi Mu, Willy Susilo, Bo Qin, and Josep Domingo-Ferrer. Asymmetric group key agreement. In *Proceedings of the 28th Annual International Conference on Advances in Cryptology: the Theory and Applications of Cryptographic Techniques*, EUROCRYPT '09, pages 153–170, Berlin, Heidelberg, 2009. Springer-Verlag. pages 48

[26] Adam Young and Moti Yung. The dark side of "black-box" cryptography, or: Should we trust capstone? In *Proceedings of the 16th Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '96, pages 89–103, London, UK, 1996. Springer-Verlag. pages 21

[27] Adam Young and Moti Yung. Kleptography: using cryptography against cryptography. In *Proceedings of the 16th annual international conference on Theory and application of cryptographic techniques*, EUROCRYPT'97, pages 62–74, Berlin, Heidelberg, 1997. Springer-Verlag. pages 21

[28] Wei Yuan, Liang Hu, Hongtu Li, and Jianfeng Chu. An efficient password-based group key exchange protocol using secret sharing. *Appl. Math. Inf. Sci*, 7(1):145–150, January 2013. pages 29, 33, 38

[29] Wei Yuan, Liang Hu, Hongtu Li, and Jianfeng Chu. Security and improvement of an authenticated group key transfer protocol based on secret sharing. *Appl. Math. Inf. Sci*, 7(5):1943–1949, September 2013. pages 29, 30