

Verificarea protocoalelor de autentificare de grup prin Scyther

Andrei Cristian
andrei.cristian1@info.uaic.ro

May 19, 2021

- 1 Introducere
- 2 Descrierea protocoalelor de autentificare de grup
- 3 Utilitarul Scyther
 - Modelul adversarului în Scyther
 - Specificarea cerințelor de securitate în Scyther
- 4 Analiza formală a protocoalelor de Tip I
 - Formalizarea cerințelor de securitate
 - Specificare problemelor dificile
- 5 Analiza formală a protocoalelor de Tip II

Introducere

Lucrarea "Verifying Group Authentication Protocols by Scyther", Huihui Yang, Vladimir Oleshchuk, and Andreas Prinz (University of Agder, Kristiansand, Norway) prezintă analiza a două protocoale complexe de autentificare de grup folosind Scyther.

Din cauza limitării utilitarului, doar un subset de proprietăți de securitate au fost verificate:

- autentificare mutuală;
- autentificare cu cheie implicată ¹;
- siguranță împotriva atacurilor de impersonare și adversarilor pasivi.

¹proprietatea în care una dintre părți este asigurată că nicio altă parte în afară de o a doua parte identificată în mod specific nu poate avea acces la o anumită cheie secretă

Scyther

- Pentru verificarea securitatii protocoalelor exista doua abordari principale: securitatea demonstrabila (eng. *provable security*) si metodele formale (eng. *formal methods*).
- **Scyther** este un utilitar de verificare formala si este conceput pentru verificarea automata a protocoalelor de securitate.
- Modelul adversarial este predefinit si anume modelul Dolev-Yao. Aceasta abordare simplifica formalizarea protocoalelor de securitate si il face mai usor de folosit pentru utilizatorii noi.
- Poate oferi clase de comportament de protocol spre deosebire de doar urmele de atac (furnizate in cazul altor utilitare).

Protocoale de autentificare de grup

- Scopul principal este imbunatatirea eficientei autentificarii pentru grupuri mari
- Relatia dintre autentificator si utilizatorii care urmeaza sa fie autentificati este unu la unu.
- In acest tip de protocol de autentificare de grup, autentificatorul **poate autentifica mai multi utilizatori in acelasi timp**.
- Daca in protocol autentificarea are acces:
 - autentificarea mutuala ar trebui sa fie satisfacuta;
 - se va stabili o cheie de sesiune de grup.

Ce urmarește lucrarea

- Extinderea lucrării [1];
- În lucrarea menționată, mărimea grupului era de 3, iar în această lucrare sunt analizate cazurile pentru grupuri ce conțin doi, trei și patru membri.
- Formalizarea protocoalelor bazate pe DLP² de tipurile I și II (tip II = autentificatorul are certificat bazat pe PKI³; tip I = autentificatorul nu are certificat) când numărul de membri din grup este $N(N \geq 3)$.
- Analiza unor noi proprietăți ale protocoalelor, precum "Alive" și "Nisynch"

²discrete logarithm problem

³public key infrastructure

Scenarii de utilizare

- ①
 - Asa cum este prezentat in figura 8, autentificatorul de tip I are o lista de prieteni, dar membrii din aceasta lista se pot sau nu cunoaste intre ei.
 - De fiecare data inainte de intalnirea grupului, autentificatorul intai selecteaza membrii grupului si apoi trebuie sa autentifice fiecare membru din acest grup.
 - Cum toti membrii s-au inregistrat deja ca prieteni ai autentificatorului, presupunem ca acestia partajeaza niste secrete cu autentificatorul inainte de autentificare.

Scenariul de utilizare pentru protocoale de tip I

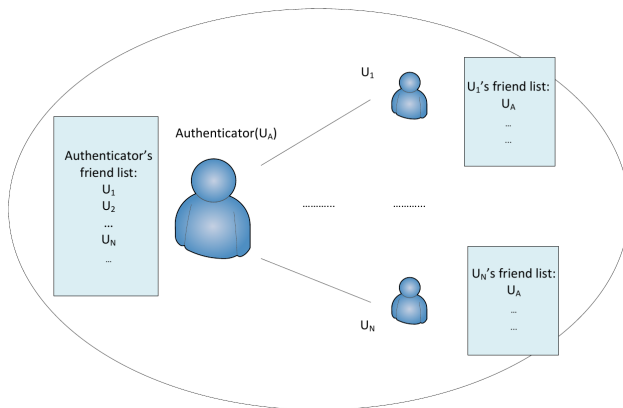


Figure: Scenariul de utilizare 1: Tipul I

Scenarii de utilizare

- 2 • În protocoalele de tip II (figura 10), autentificatorul este un server.
- Trebuie să autentifice utilizatori pe care nu îi cunoaște neapărat dinainte.
- În acest caz, serverul trebuie să dețină un certificat pentru a realiza autentificarea grupului.

Scenariul de utilizare pentru protocoale de tip II

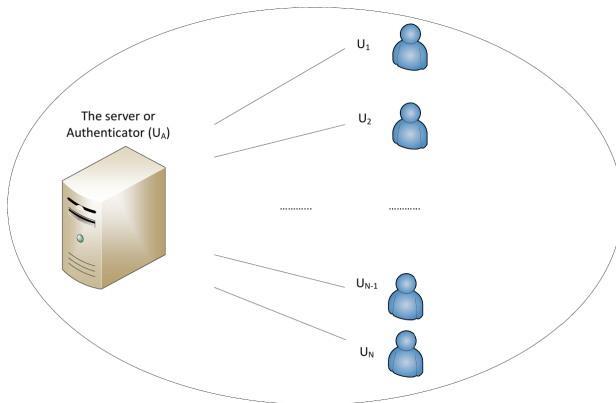


Figure: Scenariul de utilizare 2: Tipul II

Un framework general

Presupunem ca sunt N membri in grupul de utilizatori \mathbb{U} . Fluxul de mesaje al frameworkului general propus in [3] poate fi descris in urmatorii patru pasi:

- ① $U_A \rightarrow U_1 : ID_A, UID, X, C_0, MAC_A.$
- ② $U_i \rightarrow U_{i+1} : ID_i, UID, X, KP_U, C_i, MAC_i,$ unde $1 \leq i \leq N - 1.$
- ③ $U_N \rightarrow U_A : ID_N, KP_U, C_N, MAC_N.$
- ④ $U_A \rightarrow \mathbb{U} : Y, MAC'_A.$

Pentru $j \in \{A, N, U_i\}$, ID_j este identitatea lui j , UID este setul de identitati al tuturor utilizatorilor din \mathbb{U} , X este o informatie importanta pe care U_A vrea sa o transmita la tot grupul, C_k este utilizat pentru a calcula C_{k+1} , pt. $k = 0..N - 1$, MAC_j este codul de autentificare al mesajului, KP_U este setul de parametri cheie al grupului de utilizatori \mathbb{U} , iar Y contine parametrii cheie generati de

Protocoale bazate pe problema logaritmului discret

Calcularea parametrilor $C_i (0 \leq i \leq N)$, X și Y ai protocolului bazat pe DLP pentru ambele tipuri Tipul I și Tipul II.

- 1 C_0 este calculat de către U_A prin $C_0 = \xi(r) = \xi(g_A^r)$, unde $r_A \in [1, p-1]$ este un număr generat aleator, ξ este un mesaj ce va fi criptat prin algoritmul de criptare Elgamal. Similar, $U_i (2 \leq i \leq N)$ calculează $C_i = C_{i-1} \times r^{x_i} = \xi(r^{\sum_{t=1}^i x_t})$.
- 2 X este calculat ca soluție a $X \equiv V_i \pmod{k_i} (1 \leq i \leq N)$, folosind teorema chineză a resturilor (CRT), unde k_i este un secret partajat între U_A și U_i .

Protocoale bazate pe problema logaritmului discret

- Tipul I: $V_i = \{y_i \oplus K_G, y_i \oplus t_i, g^{m_i}, h_i\}$, iar $h_i = H(ID_A \oplus ID_i \oplus y_A \oplus t_i)$ si este folosit pentru autentificarea lui U_A cu U_i . Aici, y_i este un secret predistribuit intre U_A si U_i , K_G este cheia de sesiune a grupului generata de U_A , t_i este un nonce si g^{m_i} este parametrul cheie generat de U_A pentru a calcula cheia partajata intre U_A si U_i .
 - Tipul II: $V_i = SIGN_{SK_A}\{ID_A, ID_i, K_G, G^{m_i}, t_i\}$. Parametrii g^{m_i} si t_i au aceeasi semnificatie ca in cazul Tipului I. Autentificarea lui U_A este realizata prin verificarea folosind semnatura sa in loc de utilizarea lui h_i ca in cazul Tipului I.
- ③ U_A calculeaza Y prin rezolvarea $Y \equiv W_i \bmod k_i (1 \leq i \leq N)$, unde $W_i = \{ID_A, ID_i, KP_i\}$ si $KP_i = KP_U - \{G^{n_i}\}$.

Protocoale bazate pe problema logaritmului discret

- 4 Cheia sesiunii dintre U_A si U_I este calculata ca fiind $g^{m_i n_j}$, in timp ce cheia sesiunii dintre U_i si U_j ($1 \leq i, j \leq N, i \neq j$) este calculata ca fiind $g^{n_i n_j}$.

Modelul adversarului in Scyther

- Modelul adversarului in Scyther este predefinit si se bazeaza pe modelul *Dolev-Yao* [2].
- Nu trebuia sa formalizam abilitatile adversarului cand analizam protocoale.
- Adversarul (notat cu **A**) poate **intercepta mesaje** de pe canalul de comunicare si poate **invata** din mesajele pe care le are.
- Presupunem ca M este setul de cunostinte al adversarului si f este o functie prin care se exprima relatiile intre diferite elemente din M .

Modelul adversarului în Scyther

- k poate reprezenta atât o cheie simetrică, dar și asimetrică, iar k^{-1} este inversul acesteia ($k^{-1} = k$ în cazul cheii simetrice).
- Fie (t_i, t_j) reprezentarea concatenării între termenii t_i și t_j .
 - $t \in M \Rightarrow M \vdash t$: dacă t este un element al lui M , atunci A cunoaște t .
 - $M \vdash (t_1, t_2) \Rightarrow \{M \vdash t_1, M \vdash t_2\}$: dacă A cunoaște (t_1, t_2) , atunci A cunoaște ambii termeni t_1 și t_2 .
 - $\{M \vdash t_1, M \vdash t_2\} \Rightarrow M \vdash (t_1, t_2)$: dacă A cunoaște ambii termeni t_1 și t_2 , atunci A cunoaște (t_1, t_2) .
 - $\bigwedge_{1 \leq i \leq n} M \vdash t_i \Rightarrow M \vdash f(t_1, \dots, t_n)$: dacă A cunoaște toți $t_i (1 \leq i \leq n)$ și f este o funcție publică, atunci A poate calcula rezultatul funcției f cu datele de intrare t_1, \dots, t_n .

Modelul adversarului în Scyther

- $\{M \vdash t, M \vdash k\} \Rightarrow M \vdash \{t\}_k$: dacă A cunoaște mesajul t și cheia k , atunci A poate calcula mesajul criptat $\{t\}_k$.
- $\{M \vdash \{t\}_k, M \vdash k^{-1}\} \Rightarrow M \vdash t$: dacă A cunoaște mesajul criptat $\{t\}_k$ și cheia de decriptare k^{-1} , atunci A poate decripta criptotextul și obține astfel plaintextul t .
- În plus, adversarul A poate să ștergă, să creeze noi mesaje și să le insereze în canalul de comunicare.

Specificarea cerințelor de securitate în Scyther

- Evenimentul **match** poate fi folosit în două moduri diferite:
 - 1 Specificarea constrângerilor de egalitate - de exemplu codurile după evenimentul $match(p_1, p_2)$ pot fi executate doar dacă p_1 este egal cu p_2 .
 - 2 Asemănător cu '=' din limbajul de programare **C**, dacă p este o variabilă, iar v este o valoare, atunci $match(p, v)$ semnifică asignarea valorii v variabilei p .
- **claim** este folosit pentru specificarea cerințelor de securitate **Alive**, **Nisynch**, **secret** și **commitment**.
 - **Alive** este o formă de autentificare care are ca scop asigurarea ca într-adevăr partea de comunicare destinată (R) a executat niste evenimente - $claim(R, \mathbf{Alive})$.

Specificarea cerințelor de securitate în Scyther

- **Nisynch** semnifică faptul că toate mesajele primite de R sunt într-adevăr trimise de către partenerul de comunicare (sender) și au fost primite de către celălalt partener de comunicare (receiver) - $claim(R, \mathbf{Nisynch})$.
- $claim(R, \mathbf{secret}, rt)$ înseamnă că R pretinde că termenul rt să nu fie știut de către adversar.
- dacă rt este o cheie de sesiune, folosim $claim(R, \mathbf{SKR}, rt)$ pentru a specifica acest lucru.
- **Commitment** este o promisiune a unei părți din comunicare către alta parte. De exemplu, $claim(R, \mathbf{Commit}, R', t)$ înseamnă că rolul R face o promisiune t către rolul R' .
Commitment este folosit pentru a verifica protocoalele împotriva **atacurilor de uzurpare** (en. impersonation attacks).

Formalizarea cerintelor de securitate

Fie R si R' doua parti de comunicare. Se pretinde ca protocoalele bazate pe problema logaritmului discret indeplinesc urmatoarele cerinte de securitate:

① Autentificarea mutuala

- Autentificarea este calea prin care asiguram o parte a comunicarii ca aceasta comunica intr-adevar cu partea dorita. Daca autentificarea este indeplinita de ambele parti ale comunicarii, aceasta se numeste **autentificare mutuala**.
- Autentificarea autentificatorului U_A cu U_i poate fi confirmata daca h'_i este egal cu h_i . Tot grupul poate fi considerat autentificat doar daca C'_N este egal cu C_N . Se va folosi **match** pentru a verifica egalitatea dintre C_N si C'_N .

Formalizarea cerintelor de securitate

- In plus, proprietatea **Alive** este necesara pentru a sti ca intr-adevar partile de comunicare sunt cele dorite.

② Autentificarea implicita a cheii

- Daca un protocol satisface autentificarea implicita a cheii k (en. implicit key authentication), iar R cere ca aceasta cerinta de securitate sa fie indeplinita, inseamna ca R' este singura entitate care are posibilitatea de a sti cheia k .
- Vom folosi $claim(R, SKR, k)$ si $claim(R', SKR, k)$ pentru a exprima aceasta proprietate.

③ Siguranta impotriva atacurilor de uzurpare

- Atac in care adversarul se comporta sub identitatea unei parti legitime de comunicare.

Formalizarea cerintelor de securitate

- Cat timp **autentificare mutuala** tine, putem pretinde ca niciuna dintre partile de comunicare nu este uzurpata de adversar (verificam daca $h'_i = h_i$ si $C'_N = C_N$).

4 Siguranta impotriva adversarilor pasivi

- Un adversar pasiv intercepteaza mesaje de pe canalul de comunicare, le analizeaza si incearca sa afle cat mai multe informatii posibile.
- Spre deosebire de un adversar activ, nu poate sa stearga sau sa introduca noi mesaje in canalul de comunicare.
- Principalul scop este sa invete informatii **utile** din mesajele interceptate.
- In protocoalele de tipul I bazate pe DLP, cea mai utila informatie este cheia grupului (k) si cheile de sesiune (k).

Formalizarea cerintelor de securitate

- Utilizam $claim(R, SKR, K)$ pentru a exprima aceasta cerinta.

5 Furnizarea de forward secrecy si backward secrecy

- Daca un protocol furnizeaza **forward secrecy**, atunci expunerea cheilor din sesiunea curenta nu va duce la expunerea cheilor din sesiunile viitoare.
- Daca un protocol furnizeaza **backward secrecy**, atunci compromiterea cheilor din sesiunea curenta nu cauzeaza compromiterea cheilor din sesiunile anterioare.
- Cum Scyther nu permite valori cu durata lunga, in afara de cheile partajate intre doua parti, aceste doua cerinte de securitate nu sunt analizate.

Specificarea problemelor dificile

① Diffie-Hellman

- Tipul **hashfunction** este folosit pentru a declara o funcție hash sigură (funcție *one-way* - calculul inversului ei este irealizabil).
- Probleme matematice dificile, precum problema Diffie-Hellman folosită pentru a calcula cheile de sesiune, funcții hash criptografice, criptare proxy și MAC pot fi considerate o funcție hash *one-way*, deoarece adversarul definit de Scyther nu poate să îi calculeze inversul.
- Dacă două părți A și B vor să stabilească o cheie de sesiune bazată pe protocolul de schimb de chei Diffie-Hellman, atunci ei trebuie:
 - ① să genereze parametrii a și b .
 - ② să trimită g^a și g^b unul altuia.

Specificarea problemelor dificile

- ③ sa calculeze cheile lor de sesiune $(g^b)^a$, respectiv $(g^a)^b$.
- Formalizat, declarăm două **hashfunction** g and h și apoi scriem cheile de sesiune ca $h(g(b), a)$ și $h(g(a), b)$. Similar, folosim **hashfunction** H , C și MAC pentru a specifica funcții hash, criptare proxy și MAC.

② Teorema chineza a resturilor (CRT)

- În protocoalele originale [3], parametrii X și Y sunt calculați prin $X \equiv V_i \bmod k_i$ ($1 \leq i \leq N$) și $Y \equiv W_i \bmod k_i$ ($1 \leq i \leq N$) folosind CRT, unde k_i este o valoare cu termen lung partajată între U_A și U_i .
- Cum Scyther nu suportă valori cu termen lung în afara de chei simetrice/asimetrice, se va folosi o cheie simetrică între U_A și U_i pentru a simula această valoare k_i .

Specificarea problemelor dificile

3 Secrete prepartajate

- x_i ($1 \leq i \leq N$) este o alta valoare partajata pe termen lung. Ea este folosita pentru autentificarea mutuala.
- Cum cheia simetrica $k(U_A, U_i)$ este deja folosita pentru a simula k_i , x_i trebuie formalizat diferit.
- Cum x_i este o valoare cu termen lung folosita pentru autentificarea mutuala, ar trebui sa fie de ajuns ca x_i sa fi fost deja partajat intre U_A si U_i inainte de autentificarea mutuala.
- Asadar, x_i va fi inclus in X . Cum parametrii din V_i pot fi extrasi doar de U_i , aceasta asumptie este rezonabila si realistica.


Analiza formală a protocoalelor de Tip II

- Specificarea protocoalelor de tip II bazate pe DLP este similară ca în cazul celor de tip I: formalizarea problemelor dificile și a cerințelor de securitate sunt la fel, excepție făcând autentificarea mutuală.
- Comparând cu formalizarea protocoalelor de tip I bazate pe DLP, există două mari diferențe:
 - ① Când U_A trimite mesaje ce includ V_i către toți membrii grupului, în protocolul de tip I, h_i este inclus pentru autentificarea mutuală. Totuși, în protocoalele de tip II, U_A își folosește semnatura pentru autentificare.

Analiza formală a protocoalelor de Tip II

- 2 Cand utilizatorii din group primesc aceste mesaje de la U_A , ei nu mai trebuie sa verifice egalitatea lui h_i pentru a incheia autentificarea lui U_A . In schimb, ei verifica semnatura lui U_A . Aceasta proprietate poate fi asigurata prin securitatea semnaturilor bazate pe PKI, deci nu trebuie verificata aici.

Bibliografie

-  H. Yang, V. Oleshchuk, and A. Prinz, “Verifying group authentication protocols by scyther,” *J. Wirel. Mob. Networks Ubiquitous Comput. Dependable Appl.*, vol. 7, pp. 3–19, 2016.
-  D. Dolev and A. Yao, “On the security of public key protocols,” *IEEE Transactions on Information Theory*, vol. 29, no. 2, pp. 198–208, 1983. DOI: 10.1109/TIT.1983.1056650.
-  H. Yang, L. Jiao, and V. A. Oleshchuk, “A general framework for group authentication and key exchange protocols,” in *Foundations and Practice of Security*, J. L. Danger, M. Debbabi, J.-Y. Marion, J. Garcia-Alfaro, and N. Zincir Heywood, Eds., Cham: Springer International Publishing, 2014, pp. 31–45, ISBN: 978-3-319-05302-8.