Protocoale de autentificare

Prof. Dr. Adrian Atanasiu

October 21, 2014

Mecanismele de autentificare se referă atât la partenerii de dialog cât și la mesajele transmise de aceștia.

Principala deosebire între cele două tipuri: autentificarea unui mesaj nu dă nici o garanție asupra momentului creerii mesajului, pe când autentificarea unui utilizator este legată implicit de momentul solicitării acestei autentificări.

Analog, autentificarea unui utilizator nu oferă nici o informație despre conținutul mesajelor pe care le va gestiona acesta în calitate de entitate autorizată, pe când autentificarea unui mesaj oferă anumite date despre conținutul mesajului.

Autentificarea utilizatorului

A. **Controlul accesului**:
Controlul accesului acordă sau restricționează dreptul unui utilizator de a accesa anumite resurse; de asemenea, el protejează resursele, limitând accesul doar pentru utilizatorii autorizați.

B. **Autorizarea**:
Procesul prin care unui utilizator i se alocă diverse drepturi de acces. Ele includ anumite specificații: dreptul de a citi, de a scrie sau de a actualiza un anumit fișier.
De exemplu, protocoalele de plăți electronice solicită autorizarea celor care le utilizează, pentru a preveni fraudarea tranzacțiilor.
O solicitare similară apare și în cazul protocoalelor de vot electronic.

Continuare:

C. **Auditarea**:
Este recomandabil să se rețină acțiunile tuturor utilizatorilor. Astfel de arhive sunt utile în cazul unor posibile apariții de activități malițioase.

Multe atacuri asupra sistemelor de calcul provin de la persoane autorizate chiar de sistemele respective.

Autentificare versus identificare

În general există o distincție netă între noțiunile de "*autentificare*" a utilizatorului și "*identificarea*" sa.

Protocoalele de identificare se referă la situația următoare: utilizatorul respectiv a fost autentificat dar el trebuie să demonstreze că are dreptul să solicite accesul la anumite resurse.

De asemenea, noțiunile numite generic "*zero - knowledge*" se referă la protocoale teoretice de identificare.

În general, sistemele de autentificare a utilizatorilor se construiesc cu scopul de a determina

1. Ceva ce utilizatorul știe.
   De exemplu, o *parolă*.
   Multe sisteme solicită CNP-ul sau data nașterii pentru a autentifica un utilizator.
2. Ceva ce utilizatorul posedă.
   De obicei este vorba de un lucru – fizic sau electronic – care aparține utilizatorului.
3. Ceva ce utilizatorul este (sau îl caracterizează).
   Acesta este principiul *biometric* de măsurare a anumitor proprietăți biologice ale utilizatorului.
   Aceste proprietăți biometrice pot fi statice (măsurători de amprente, retină etc) sau dinamice (analiza vocii, recunoașterea scrisului de mână etc).

Parole

O parolă este o secvență alfanumerică cunoscută doar de utilizator (care se autentifică) și de sistemul care asigură autentificarea.

Uneori este posibil ca sistemul să nu cunoască parola, dar să o poată deduce din anumite informații pe care le deține despre utilizator.

Din motive de securitate, parolele trebuie să fie secvențe pe care *Alice* le poate memora sau – eventual – deduce ușor.
În plus, ele trebuie să fie greu de "ghicit" de către orice altă entitate exterioară.

Sistemul de calcul păstrează parolele sub o formă criptată sau ca amprente, pentru a evita accesul în cazul unui atac reușit.

Vulnerabilități

In 1979, Morris și Thompson au analizat 3289 parole colectate aleator de la angajații care utilizau tehnică de calcul (în special IBM). 2831 (86%) din acesta erau vulnerabile deoarece:

- 15 erau formate dintr-un singur caracter ASCII
- 72 erau formate din 2 caractere ASCII
- 464 erau formate din 3 caractere ASCII
- 477 erau formate din 4 caractere alfanumerice
- 706 erau formate din 5 litere, de același tip
- 605 erau formate din 6 litere, toate mici.

Alte vulnerabilități ale parolelor includ:

- *Cuvinte uzuale.* Engleză, română sau alte limbi.
- *Nume cunoscute.* Nume de vedete, animale, prieteni, membri ai familiei, porecle.
- *Informații ușor de obținut.* Date de naștere, numere de telefon, numărul mașinii etc.
- *Secvențe de tastatură.* Ceva de genul "qwerty".
- *Permutări ale celor de sus.* De obicei scrieri inverse (în oglindă).

Parole one-time

Asigură securitate sporită în fața atacurilor prin forță brută.
Strategii de a asigura astfel de parole:

A. Unele sisteme oferă utilizatorului o listă de parole.
Atunci când utilizatorul folosește o parolă, sistemul verifică dacă este în această listă.

Variantă: o tabelă în care intrările sunt perechi *întrebare/răspuns*.
Aici utilizatorul solicită autentificarea, sistemul alege aleator o întrebare și așteaptă răspunsul.

Parole actualizate secvențial

Inițial există o singură parolă (secretă).

În timpul autentificării folosind parola α, utilizatorul generează și transmite sistemului o nouă parolă

$$\alpha' = e_K(\alpha)$$

unde K este o cheie derivată din α.

Pentru următoarea comunicare, α este înlocuită cu α' .

Metoda devine dificilă atunci când apar întreruperi de comunicație.

Parole bazate pe funcții neinversabile

Par cele mai eficiente tipuri de parole *one - time*.

Cea mai cunoscută strategie de construcție de astfel de parole este schema Lamport.

Schema Lamport

Este folosită o funcție neinversabilă ϕ.
În faza de inițializare, *Alice* efectuează următoarele operații:

1. Alege un mesaj secret w și un număr n de autentificări bazate pe w (uzual $n = 100$ sau $n = 1000$).
2. Transferă lui *Bob* printr-un canal sigur valoarea $w_0 = \phi^n(w)$.
3. *Bob* inițializează un counter $i_{Alice} := 1$.

În timpul autentificării pentru deschiderea sesiunii cu numărul i, se procedează astfel:

1. *Alice* calculează $w_i = \phi^{n-i}(w)$ și trimite lui *Bob* tripletul
$$(Alice, i, w_i)$$
(calculul lui w_i se poate face la fiecare nouă sesiune, sau se poate deduce dintr-o tabelă construită inițial, odată cu operațiile efectuate la determinarea lui w_0).
2. *Bob* verifică dacă $i = i_{Alice}$ și dacă $\phi(w_i) = w_{i-1}$.
Dacă ambele relații sunt verificate, *Bob* acceptă autentificarea, salvează w_i pentru verificarea următoare și incrementează contorul: $i_{Alice} := i_{Alice} + 1$.

Variantă a protocolului Lamport

Alice deține o parolă α dată de *Bob*.
Atunci când dorește autentificarea, *Alice* trimite o pereche

$$(r, h(r\|\alpha))$$

unde r este o secvență binară, iar h este o funcție de dispersie criptografică.

Bob face verificarea calculând $h(r\|\alpha)$ și compară rezultatul cu al doilea element al perechii primite.

Pentru a elimina atacurile unui adversar activ, r trebuie să fie un element de tip *nonce*.

Măsurători biometrice

Folosesc la autentificarea unor caracteristici fizice ale unei persoane.
În general ele nu sunt folosite drept parole, deoarece – odată compromise – nu pot fi înlocuite.
Cele mai utilizate măsurători biometrice sunt:

- Recunoașterea vocii,
- Dinamica semnăturii,
- Amprente,
- Geometria mâinii,
- Scanarea retinei,
- Scanarea irisului,
- Modelul facial,
- Alte caracteristici specifice.

La construirea unui sistem de autentificare cu măsurători biometrice trebuie ținut cont de:

- timpul necesar efectuării acestor măsurători,
- prețul aparaturii,
- gradul de acceptare al utilizatorului de a se expune măsurătorilor biometrice,
- ratele de eroare privind falsa - acceptare (a unui alt utilizator decât cel legal),
- falsa - respingere (rejectarea utilizatorului legal).

Rata de eroare

Tehnica	**Rata de eroare**
Voce (analiză Alpha)	3%
Voce (analiză ECCO)	2%
Semnătură	2%
Scanarea retinei	0.4%
Geometria mâinii	0.1%
Amprente	9% falsa - respingere,
	0% falsa - acceptare

Caracteristici	**Cea mai bună**	**Cea mai slabă**
Acceptare	Mâna	Voce
Falsa - respingere	Mâna	Amprente
Falsa - acceptare	X	Voce
Mulțime detalii	X	Voce, semnătura
Dificultatea imitării	Retina	Voce, semnătura
Cost	Voce	Retina

X = mâna, retina, amprente.
Datorită ratei mari de eroare, măsurătorile biometrice nu constituie un avantaj față de parole.

De obicei aceste două tipuri de autentificare se folosesc combinat (deci sistemele verifică atât ceea ce utilizatorul "*este*" cât și ceea ce utilizatorul "*știe*").

Protocoale de autentificare a mesajelor

Este separat de autentificarea utilizatorului care trimite mesajul (autentificarea ambelor entități se realizează folosind semnătura electronică).

De asemenea, autentificarea unui mesaj nu înseamnă totdeauna și integritatea lui (deși în majoritatea protocoalelor, acest lucru se subînțelege).

Sunt domenii de securitate a informației dedicate special problemelor de autentificare și integritate a mesajelor.
Exemplu: Watermarking și Fingerprint.

MAC

Definiție

Un cod de autentificare a mesajului (MAC) este o combinație între o funcție de compresie și o cheie.

Este folosit pentru a autentifica mesajul și a-i certifica integritatea.

Codurile de autentificare a mesajelor sunt utilizate pe scară largă în protocoale legate de comunicarea pe Internet (IPSec sau SSL/TLS).

Atunci când *Alice* trimite lui *Bob* un mesaj α (criptat sau nu), va construi un cod $MAC(\alpha)$ folosind o cheie și funcție de compresie cunoscute doar de ambii parteneri.
Va trimite perechea

$$(\alpha, MAC(\alpha))$$

La recepția unei perechi (α, x), *Bob* calculează $MAC(\alpha)$ și vede dacă acesta coincide cu x.

În caz afirmativ, el va considera mesajul α ca fiind autentic.

Definiție

O pereche (α, x) cu $x = MAC(\alpha)$ se numește "pereche validă".

Atât *Alice* cât și *Bob* pot calcula un *MAC* valid pentru un mesaj α.

HMAC

Dacă drept funcții de compresie sunt folosite funcții de dispersie, codurile de autentificare a mesajelor se numesc coduri *HMAC* (Hash MAC).

Exemplu

Unul din primele HMAC-uri a fost propus de IBM pentru mesajele trimise pe Internet.

Fie K cheia secretă folosită; ea este partajată în două subchei $K = (k_1, k_2)$.
Pentru fiecare bloc de text clar α, codul de autentificare va fi

$$MAC(\alpha) = MD5(k_1 \| MD5(k_2 \| \alpha))$$

Standardul RFC 2104

Fie h o funcție de dispersie criptografică care procesează mesaje de n octeți și produce rezumate de p octeți (dacă h este $SHA1$, atunci $n = 64$ și $p = 20$).

Se mai definește un parametru t $(4 < t < p)$ care reprezintă numărul de octeți din $HMAC$.

Dacă x este blocul de intrare și K este cheia folosită, calculul lui $HMAC(x)$ este:

1. Dacă $|K| > 8n$, se înlocuiește K cu $h(K)$.
2. Se completează K la dreapta cu biți $'0'$ până ajunge la n octeți.
3. Se calculează

$$A = h(K \oplus opad \| h(K \oplus ipad \| x))$$

unde
$ipad, opad \in \{0,1\}^{8n}$, $ipad = (36 \ldots 36)_{16}$, $opad = (5C \ldots 5C)_{16}$.

4. $HMAC_K(x)$ este format din primii t octeți din A.

Criptanaliza

Obiectivul unui atac este de a produce o pereche (α, x) validă pentru o cheie K (necunoscută dar fixată).

Printr-un atac cu text clar ales, *Oscar* va solicita *MAC*-uri pentru mesajele $\alpha_1, \alpha_2, \ldots, \alpha_n$ alese de el.

Deci, el va dispune de o listă de perechi valide

$$(\alpha_1, x_1), (\alpha_2, x_2), \ldots (\alpha_n, x_n)$$

generate cu o cheie necunoscută K.
Ulterior, când *Oscar* va emite o pereche (α, x), se presupune că $x \notin \{x_1, x_2, \ldots, x_n\}$.

Dacă perechea emisă este validă, spunem că ea este un *fals*.
Un MAC pentru care probabilitatea de a obține un fals este neglijabilă se numește *MAC sigur*.

Securitatea standardului RFC 2104 este asigurată de:

Teoremă

Fie $h : \{0,1\}^* \longrightarrow \{0,1\}^p$ o funcție de dispersie criptografică. Fiind date două chei $K_1, K_2 \in \{0,1\}^p$, considerăm algoritmul MAC definit prin

$$MAC_{K_1,K_2}(x) = h(K_2 \| h(K_1 \| x))$$

Dacă aplicația MAC_{K_2} definită

$$MAC_{K_2}(x) = h(K_2 \| x)$$

este un algoritm MAC sigur pentru mesaje x, $|x| = p$, atunci MAC_{K_1,K_2} este un algoritm MAC sigur pentru mesaje de lungime arbitrară.

CBC – MAC

Cea mai simplă modalitate de a construi un cod de autentificare al unui mesaj α este de a folosi un sistem de criptare simetric implementat în modul *CBC* și de a utiliza ultimul bloc obținut prin criptarea lui α drept $MAC(\alpha)$.

1. Textul clar x se împarte în blocuri de lungime fixată: $x = \alpha_1\alpha_2 \ldots \alpha_n$.
2. Se construiește secvența $\beta_1, \beta_2, \ldots, \beta_n$ după formula

$$\beta_i = e_K(\beta_{i-1} \oplus \alpha_i) \qquad (i \geq 1)$$

   unde $\beta_0 = 00 \ldots 0$, iar e_K este funcția de criptare cu cheia K.
3. $MAC(x) = \beta_n$.

Avantaje: Metoda este foarte rapidă și ușor de realizat.

Atac asupra *CBC – MAC*

Să presupunem că știm trei perechi valide

$$(x_1, c_1),\ (x_2, c_2),\ (x_3, c_3).$$

unde x_1 și x_3 sunt mesaje de aceeași lungime, iar x_1 este un prefix propriu al lui x_2; deci $x_2 = x_1 \| \alpha \| x_2'$, unde α este un bloc.
Blocul criptat obținut – în calculul lui c_2 – după criptarea lui α este $e_K(\alpha \oplus c_1)$.
Să notăm $\alpha' = \alpha \oplus c_1 \oplus c_3$ și $x_4 = x_3 \| \alpha' \| x_2'$.
În calculul MAC-ului c_4 (pentru x_4), după criptarea lui α' se obține

$$e_K(c_3 \oplus \alpha') = e_K(\alpha \oplus c_1),$$

după care criptările pentru c_2 și c_4 merg similar (urmând după blocul α').
Deci, în final $c_4 = MAC(x_4) = c_2$ și se poate construi o pereche validă (x_4, c_2).

EMAC

Codul de autentificare *EMAC* (*Encrypted MAC*) este obținut prin criptarea cu altă cheie K' a ultimului bloc criptat β_n.
Securitatea lui este egală cu rezistența lui *CBC* la un atac bazat pe paradoxul nașterilor.
Într-adevăr, să presupunem că *Oscar* obține o coliziune: două perechi valide (x_1, c), (x_2, c) cu același *MAC*.
El ia un mesaj arbitrar x_3' și solicită un cod de autentificare pentru mesajul $x_3 = x_1 \| x_3'$.
Să presupunem că primește $MAC(x_3) = c'$.
Atunci $(x_2 \| x_3', c')$ este o pereche validă.

Pentru a rezista la un astfel de atac, modul *CBC* de implementare utilizat pentru construirea unui *EMAC* trebuie să fie similar unei funcții de dispersie criptografică.

OMAC

One-key CBC MAC este un standard care operează cu mesaje a căror lungime nu este totdeauna un multiplu al lungimii blocului de criptare.

Lucrează cu un sistem simetric $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$, având lungimea blocurilor criptate egală cu p, o cheie $K \in \mathcal{K}$, o familie $\mathcal{A}$ de constante și două constante $C_1, C_2 \in \mathcal{A}$.
Se mai definește o valoare t $(t < p)$ ca fiind lungimea codului de autentificare.
Fie familia de funcții

$$\mathcal{H} = \{H_L \mid L \in \mathcal{C}, \quad H_L : \mathcal{A} \longrightarrow \mathcal{C}\}$$

Să considerăm textul clar $x = \alpha_1 \| \alpha_2 \| \ldots \| \alpha_n$ cu $|\alpha_i| = p$ pentru $i < n$, $\quad |\alpha_n| \leq p$.

Construcție

1. Fie $L = e_K(00\ldots0)$.
   Se determină $H_L(C_1)$ și $H_L(C_2)$.
2. Dacă $|\alpha_n| < p$, α_n se completează cu bitul $'1'$ urmat – eventual – de un șir arbitrar de biți, până ajunge la lungimea p.
   Spunem că α_n "a fost completat".
3. $\alpha_n := \begin{cases} \alpha_n \oplus H_L(C_1) & \text{dacă } \alpha_n \text{ nu a fost completat} \\ \alpha_n \oplus H_L(C_2) & \text{dacă } \alpha_n \text{ a fost completat} \end{cases}$
4. Se determină CBC MAC-ul pentru $\alpha_1 \| \alpha_2 \| \ldots \| \alpha_n$.
5. $OMAC(x)$ constă din primii t biți din acest MAC.

Implementare *OMAC*1

Se consideră $\mathcal{P} = \mathcal{C} = \{0,1\}^p$, $(p = 64, 128)$. Toate toate operațiile se efectuează în $GF(2^p)$, inel structurat astfel: $\alpha = a_1 a_2 \dots a_p \in \mathbb{Z}_2^p$ este asociat cu polinomul

$$\alpha(X) = a_1 X^{p-1} + \dots + a_{p-1} X + a_p \in GF(2^p)$$

Operațiile pe $GF(2^p)$ sunt definite:
- adunarea $\alpha + \beta = \gamma$ este un *XOR* bit-cu-bit.
- înmulțirea $\alpha' = \alpha \cdot X$ se face astfel:

1. Se ia vectorul α.
2. Se elimină primul bit din stânga și se inserează la sfârșit bitul $'0'$; fie β noul vector.
3. Dacă bitul eliminat a fost $'1'$, atunci

$$\alpha' = \begin{cases} \beta \oplus 00\dots1B & \text{dacă } p = 64 \\ \beta \oplus 00\dots87 & \text{dacă } p = 128 \end{cases}$$

altfel $\alpha' = \beta$.

$H_L(x) = L \cdot x$,
$C_1 = 00 \ldots 2$,
$C_2 = 00 \ldots 4$
(în $GF(2^p)$, C_1 corespunde polinomului X, iar C_2 – polinomului X^2).

Deci

$$H_L(C_1) = L \cdot X, \quad H_L(C_2) = H_L(C_1) \cdot X.$$

Modul autentificat de operare

Un *MAC* asigură autentificarea și integritatea unui mesaj.
Nu s-a pus problema confidențialității, mesajul în discuție fiind un text clar.

Uneori însă este nevoie ca mesajele criptate să fie protejate prin algoritmi de autentificare și integritate.
Acest lucru este realizat de obicei combinând operația de criptare cu cea de construire a unui MAC, procedeu numit *Authenticated Mode of Operation*.

Protocol frecvent folosit: *CCM* (Counter with CBC-MAC) – o combinație cu *AES* (sau alt sistem simetric de criptare pe blocuri de 128 biți).

CCM

Fie α un mesaj.
Se determină $T = MAC(\alpha)$ și apoi se criptează $T \| \alpha$.
Pentru construcția unui *CCM* folosesc:

- O cheie K a sistemului de criptare bloc pe 128 biți.
- Doi parametri:
  M – mărimea (în octeți) a lui T; număr par din intervalul $[4, 16]$;
  L – mărimea (în octeți) a zonei care codifică lungimea mesajului α; număr din $[2, 8]$.
  M și L sunt codificate pe câte 3 biți fiecare, anume reprezentarea în binar a lui $(M-2)/2$ respectiv $L-1$.
  Valoarea $L = 1$ este rezervată.

- Un nonce N de $15 - L$ octeți.
- O valoare 'a' de autentificare suplimentară (de exemplu un număr într-o secvență dintr-o sesiune de lucru, sau headerul unui pachet de date).
- $flag_1$: un octet definit

$$flag_1 = 0 \| adata \| M \| L$$

unde *adata* este un bit setat pe $'0'$ dacă și numai dacă $|a| = 0$.

În prima etapă se calculează un CBC MAC pentru α:

1. Se determină blocul (de 128 biți)
$$B_0 = flag_1 \| N \| |\alpha|$$
2. Se descompune α în blocuri de 128 biți: $\alpha = B_1 \| B_2 \| \ldots \| B_n$ (eventual ultimul bloc se completează cu zero la sfârșit).
3. Dacă $adata = 1$, se construiesc câteva blocuri $B_0^1, \ldots B_0^r$ formate din $|a|$ urmat de a, completate apoi cu 0 până la un multiplu de 128 biți.
4. Se calculează CBC MAC-ul mesajului
$$B_0 \| B_0^1 \| \ldots \| B_0^r \| B_1 \| \ldots \| B_n$$
5. Se rețin în T primii M octeți ai rezultatului.

În etapa a doua (de criptare):

1. Se construiesc blocurile (de numărare) $A_0, A_1, \ldots$ definite

$$A_i = flag_2 \| N \| i$$

unde numărul i este codificat pe L octeți, iar $flag_2$ este un octet cu L pe primii 3 biți și 0 în rest.
Numărul acestor blocuri depinde de $|\alpha|$.

2. T se criptează în maniera sistemelor fluide de criptare (se face un XOR între T și primii M octeți din $e_K(A_0)$).
3. α se criptează prin XOR-are cu primii $|\alpha|$ octeți din $e_K(A_1) \| e_K(A_2) \| \ldots$
4. Mesajul final este concatenarea celor două texte criptate obținute anterior.

Sfârșit