UNIUNEA EUROPEANĂ
GUVERNUL ROMÂNIEI MINISTERUL MUNCII, FAMILIEI ŞI PROTECŢIEI SOCIALE AMPOSDRU
Fondul Social European POSDRU 2007-2013
Instrumente Structurale 2007-2013
ORGANISMUL INTERMEDIAR REGIONAL PENTRU POS DRU REGIUNEA BUCUREŞTI ILFOV

Proiect cofinanțat din Fondul Social European prin
Programul Operațional Sectorial Dezvoltarea Resurselor Umane 2007-2013.
"Investește în oameni!"

Cap. 5. Protocoale Criptografice.
Cap. 5.1. Protocoale de autentificare.
Autentificarea informatiei, autentificarea entitatilor si schimburi autentificate de cheie secreta. Principii constructive: password based authentication, one-time passwords, challenge-response, zero-knowledge.

Obiectivul Autentificarii

- **Importanta**:

1) **cel mai frecvent intalnit/impus** obiectiv de securitate – folosim protocoale de autentificare aproape zilnic: citirea unui email, plata cu o carte de credit, simple convorbiri telefonice
2) **adauga valoare informatiei** – informatia nu are valoare atata timp cat nu exista o garantie asupra sursei sale de provenienta

Obiective distincte in autentificare

- Obiectivul autentificarii poate fi descompus in trei obiective distincte:
  1) **Autentificarea Informatiei**
  2) **Autentificarea Entitatilor (Identificare)**
  3) **Schimb autentic de cheie secreta** (doi participanti intra in posesia unei chei secrete cu garanti asupra sursei de la care provine cheia)

Proiect cofinanțat din Fondul Social European prin Programul Operațional Sectorial Dezvoltarea Resurselor Umane 2007-2013. "Investește în oameni!"

Tipuri de atacurilor asupra protocoalelor de autentificare

- **Retransmisie** a mesajului (**replay**) – comun in autentificarile bazate pe parole,
- **Pre-play**, varianta atacului replay – de exemplu asupra autentificarii S/Key
- **Man-in-the-middle** – adversarul (exemplu clasic protocolul de schimb de cheie Diffie-Hellman)
- **Sesiuni paralele** – doua sau mai multe sesiuni sunt rulate in paralel, adversarul extragand din una din ele raspunsuri pentru alta
- Atacuri prin **reflexie** – vezi slide protocoale challenger-response cu chei simetrice
- Atacuri de **suprapunere** (**interleaving**) – tot prin dezvoltarea de sesiuni paralele
- Atacuri datorate **tipurilor de date gresite** – confuzii intre identitati de participant si nonce etc.
- Atacuri datorate **omisiei numelor participantilor** – prin omisia numelor participantilor, mesaje din sesiunea cu un participant pot fi folosite in sesiuni cu alt participant
- Atacuri datorare **folosirii gresite a primitivelor criptografice** – se datoreaza folosirii unei primitive criptografice cu prezumtia ca asigura si alt obiective decat cel care il asigura (de exemplu multi cred ca functiile de criptare asigura si integritate)

Autentificarea Informatiei

- **Autentificarea Informatiei** (Autentificarea sursei informatiei) se refera la o **garantie asupra sursei de provenienta a informatiei**.
- Este in stransa legatura cu obiectivul integritatii informatiei pe care il implica. Exista diferente semnificative:
  - autentificarea obliga la o garantie asupra sursei de unde provine informatia iar integritatea nu
  - autentificarea implica o comunicare iar integritatea nu (date care nu se afla in tranzit pot fi garantate ca integritate dar nu ca autenticitate pentru ca puteau fi stocate de catre oricine),
  - autenticitatea implica o prospetimea informatiei (orice informatie care poate fi sursa a unui replay are integritate dar nu autenticitate).
- In urma unui schimb autentic de informatie trebuie sa poata fi stabilite: identitatea celui care a trimis informatia, integritatea informatiei, actualitatea informatiei respective.

Proiect cofinanțat din Fondul Social European prin
Programul Operațional Sectorial Dezvoltarea Resurselor Umane 2007-2013.
"Investește în oameni!"

Functii criptografice utilizate in asigurarea autenticitatii informatiei

- **Coduri de autentificare a mesajelor** MAC (Message Authentication Code) - $MAC_k(m)$ - cod de autentificare a mesajului m calculat cu cheia k :
- **Garanteaza autenticitatea** unui mesaj
- Se **construiesc peste o functie hash** (in general MD5 sau SHA1, cu toate ca ambele au un nivel de securitate destul de scazut) in conjunctie cu o cheie secreta
- In practica se foloseste **HMAC** si **NMAC** propuse de Mihir Bellare, Ran Canetti, Hugo Krawczyk in lucrarea http://www-cse.ucsd.edu/users/mihir/papers/kmd5.pdf
- **Semnaturi Digitale** – $Sig_A(M)$ – semnatura digitala a participantului A asupra mesajului M:
- Pot fi vazute ca **functionalitate aditionala a criptosistemelor cu cheie publica**
- **Garanteaza non-repudierea**, oferind astfel si o garantie asupra sursei mesajului (autenticitate)
- Se construiesc pe sisteme criptografice asimetrice prin **"inversarea" rolului cheii publice si private**, astfel se foloseste cheia privata pentru a semna mesaje iar cheia publica pentru a verifica semnatura
- **Pot fi construite si pe functii criptografice simetrice** (semnaturi digitale one-time)
- Observatie: aceste functii criptografice garanteaza autenticitatea informatiei fara a aduce garantii de timp (asupra actualitatii informatiei), in momentul in care aceste primitive sunt incluse in protocoale de autentificare trebuie asigurata si actualitatea informatiei prin parametrii varianti in timp (contor, nonce, timestamp)

Coduri MAC vs. Semnaturi Digitale

- Codurile MAC nu pot fi vazute ca si substituent pentru Semnaturi Digitale si nici invers deoarece nu raspund aceluiasi obiectiv de securitate
- Totusi, deoarece ambele pot fi utilizate pentru garantarea autenticitatii, o comparatie este utila

- Coduri MAC:
- (+) Usor de calculat, implica doar operatii simple (doar putin mai costistisitoare ca o functie hash)
- (+) Dimensiune redusa (la nivelul unei functii hash)
- (-) Utilizeaza chei secrete
- (-) Pentru comunicarea multiaterala intre n entitati numarul de chei devine n(n-1)/2

- Semnaturi Digitale:
- (-) Greu de calculat, implica operatii aritmetice complexe (in grupuri de intregi)
- (-) Dimensiune ridicata (100, 1000 de biti)
- (+) Utilizeaza chei publice
- (+) Este suficienta o singura cheie publica pentru o entitate iar cu aceasta un mesaj semnat poate fi verificat de oricare alta entitate

Proiect cofinanțat din Fondul Social European prin Programul Operațional Sectorial Dezvoltarea Resurselor Umane 2007-2013. "Investește în oameni!"

O apreciere cantitativa asupra cerintelor computationale

Cryptographic function		CPU		
		Intel Centrino 1.7 GHz	Intel Dual Core 1.6 GHz	Intel Core Duo 6600 2.4 GHz
Modular exponentiation, basic operation for a digital signature (module and exponent size in right column)	512	7.8×10^{-3} s	6.1×10^{-3} s	3.1×10^{-3} s
	1024	48.4×10^{-3} s	44.6×10^{-3} s	20.3×10^{-3} s
	2048	359.4×10^{-3} s	323.8×10^{-3} s	153.2×10^{-3} s
Mac with SHA1	160	0.00859×10^{-3} s	0.00812×10^{-3} s	0.00406×10^{-3} s
Mac with MD5	128	0.00579×10^{-3} s	0.00354×10^{-3} s	0.00219×10^{-3} s
Sha-1	160	0.00281×10^{-3} s	0.00212×10^{-3} s	0.00109×10^{-3} s
Sha-256	256	0.0086×10^{-3} s	0.00592×10^{-3} s	0.00282×10^{-3} s
Sha -384	384	0.01359×10^{-3} s	0.01234×10^{-3} s	0.00579×10^{-3} s
Sha-512	512	0.02625×10^{-3} s	0.02324×10^{-3} s	0.01141×10^{-3} s
MD5	128	0.00156×10^{-3} s	$9.5E\text{-}4 \times 10^{-3}$ s	$4.6E\text{-}4 \times 10^{-3}$ s

Autentificarea Entitatilor (Identificare)

- **Autentificarea Entitatilor** (Identificare, Verificarea Identitatii) se refera la o **garantie asupra identitatii unui participant** la comunicare (poate fi vazuta ca un caz de autentificare a informatiei in care identitatea unei entitati este informatia ce trebuie autentificata)

Baza autentificarii entitatilor

- Autentificarea unei entitati se poate baza pe:

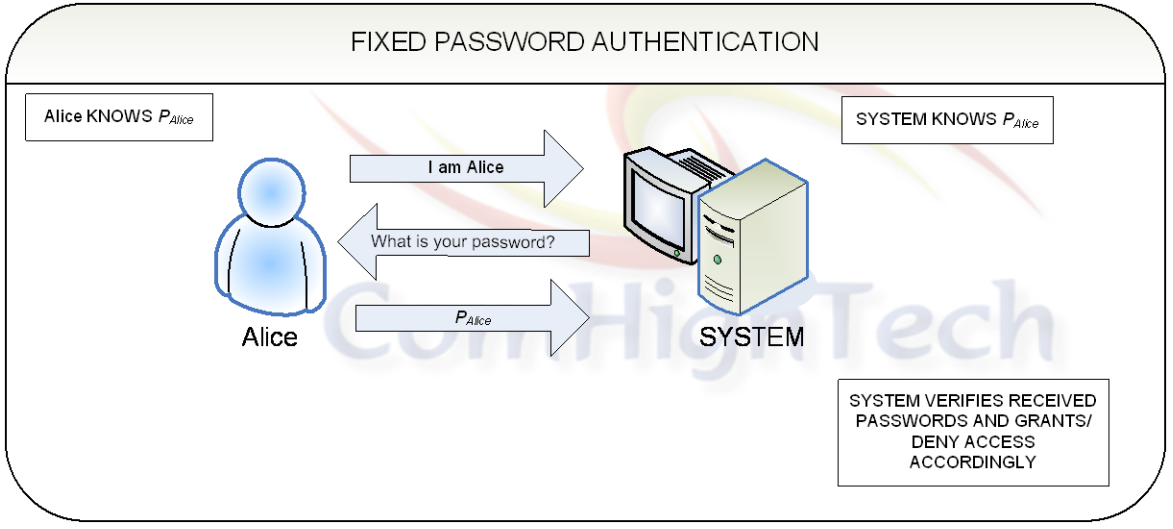
1) **Ceva cunoscut** – in general un secret, de exemplu: o parola, un cod PIN
2) **Ceva imanent (care tine de entitate)** – o trasatura biometrica: amprenta, retina, modul in care un individ tasteaza
3) **Ceva detinut** – un accesoriu fizic, de exemplu o cheie, un card magnetic, un smart-card

Caracteristici ale unui protocol de identificare ([Menezes et. al., p. 387])

- **Reciprocitate** – identificarea poate fi unilaterala, sau bilaterala (mutuala)
- **Eficienta Computationala** – intensitatea computationala a operatiilor necesare protocolului (operatiile cu primitive simetrice si fara cheie sunt ieftine iar cele cu primitive asimetrice de 10, 100 sau 1000 de ori mai scumpe)
- **Eficienta in Comunicare** – numarul de runde de comunicare si dimensiunea mesajelor vehiculate (cata informatie este necesara pentru a demonstra identitatea cuiva)
- **Implicarea real-time a unei parti terte** – uneori este necesara interventia unei parti terte pentru a putea efectua o identificare (de exemplu in cazul scenariilor bazate pe functii simetrice, pentru comunicarea intre n entitati sunt necesare n(n-1)/2 chei distincte, dar daca exista o parte de incredere o singura cheie partajata cu aceasta poate fi suficienta, in acest caz numarul de chei devine n pentru partea de incredere si 1 pentru participanti)
- **Natura implicarii partii terte** – in partea terta trebuie avuta incredere ca pastrator al confidentialitatii (securitate absoluta) sau doar asupra autenticitatii furnizata (securitate functionala)
- **Natura garantiilor de securitate** – pe ce se bazeaza securitatea?
- **Stocarea secretelor** – unde trebuie stocate secretele: pe un dispozitiv fizic, memorate etc.

Autentificarea bazata pe parole (Autentificare slaba)

- **Cel mai frecvent intalnite** si ofera si **cel mai scazut nivel de securitate** (totusi suficient pentru o buna parte din aplicatii)
- **Principiu**: autentificarea se face prin **dezvaluirea unui secret (numit parola)** pentru a demonstra identitatea unei entitati



Modul de stocare al parolelor

- **In fisiere de parole** – presupune stocarea parolei in plaintext intr-un fisier de parole. Fisierul este protejat la citire si scriere. Dezavataj: nu prezinta securitate in fata utilizatorilor privilegiati pentru a citi fisierul
- **In fisiere "criptate"** – presupune stocarea parolei intr-un fisier "criptat", in acest caz fisierul trebuie protejat doar la scriere.

- **Observatie**: termentul de fisier "criptat" pentru memorarea parolelor ("encrypted" password files) este pastrat in limbaj din ratiuni istorice, in realitate nu se foloseste o functie de criptare pentru stocarea parolelor, ci o functie one-way (de exemplu hash) pentru a face ca parola sa nu fie recuperabila din datele aflate in fisier, dar sa fie verificabila pe baza acestora

Parole fixe - Avantaje

- **Usor de implementat** (ca protocol de autentificare)
- Sunt **usor de memorat** pentru oameni

Parole fixe - Dezavantaje

- Pot fi **furate (observate)** si utilizate de un adversar
- Sunt **stocate atat de partea utilizatorului cat si a sistemului**
- Utilizatorii aleg in general parole cu **entropie scazuta**

Atacuri asupra autentificarilor bazate pe parola

- **Retransmisia (replay)** – presupune interceptarea (observarea) parolei si retransmiterea ulterioara a acesteia – **Aparare: parole one-time.**
- **Cautari exhaustive** - deoarece utilizatorii aleg parole de entropie scazuta, un potential adversar poate incerca toate parolele posibile. Cautarile exhaustive pot fi **on-line** (incearca diverse parole la login) sau **off-line** (analiza asupra unui fisier "criptat" de parole) – **Aparare: parole de entropie ridicata, politici de securitate.**
- **Atacuri de tip dictionar precalculat** – deoarece utilizatorii au obiceiul de a utiliza ca parole cuvinte (dintr-un dictionar), un adversar poate calcula off-line parole corespunzatoare tututor cuvintelor din dictionar ca apoi sa verifice daca in fisierul de parole apare o corespondenta. **Aparare: salting**.

Proceduri de intarire a parolelor

- **Politici de securitate** - obligarea la:

1) **Pentru a creste entropia**: o dimensiune minima a parolei (de ex. 10 caractere), utilizarea a minim 3 caractere din fiecare set (alfa-numerice, non-alfa-numerice, upper/lower-case)
2) **Pentru a reduce riscul unui atac**: schimbarea unei parole la intervale fixe de timp (de ex. 1 luna), limitarea numarului de incercari in cazul introducerii unei parole gresite

- **Cresterea intensitatii functiei de verificare a parolelor** – pentru a ridica necesitatile de calcul la o cautare exhaustiva poate fi crescuta intesitatea functiei de "criptare" a parolei (de exemplu iterarea functiei de n ori)
- **Adaugarea unor valori arbitrare (salt)** – pentru a opri atacuri de tip dictionar parola poate fi concatenata cu k biti aleatori, numiti salt, in acest caz dimensiunea dictionarului creste de 2^k ori, valoarea de salt se pastreaza alaturi de parola (Atentie! Aceasta metoda nu creste entropia parolei)
- **Utilizarea frazelor in locul unor simple cuvinte** – un caracter din limba curenta are cam 1-1.5 biti de entropie, utilizarea unei fraze suficient de lungi poate duce la o parola destul de rezistenta.

Autentificare cu parole de unica folosinta (one-time passwords)

- Pentru **a preveni** atacurile de tip retransmisie o solutie este utilizarea parolelor de unica folosinta (**one-time passwords**)
- Exista **doua solutii rudimentare:**

1) **Utilizarea unei liste de parole** – se foloseste o lista de *t* parole, fiecare valabila la o singura utilizare (parolele pot fi utilizate secvential sau nu)

2) **Utilizarea unor parole actualizate secvential** – la fiecare autentficare, utilizatorul schimba parola veche cu cea noua

Schema Lamport – schema de generat parole one-time

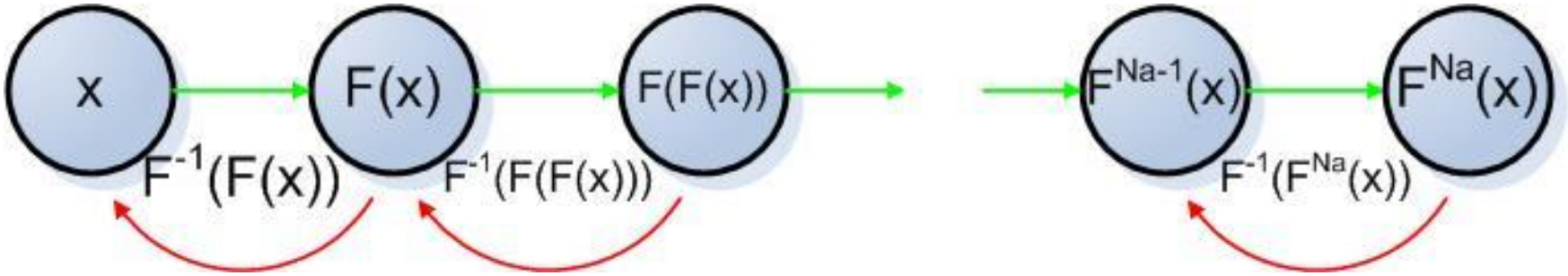
- Prima solutie avansata de generare a parolelor one-time
- **Avantaje** ale schemei Lamport:

1) **Nu se stocheaza secrete de partea sistemului**
2) Fiecare parola este **valabila doar la o singura utilizare**
3) Parolele au **entropie ridicata**
4) Poate fi **utilizata pe canale nesigure**

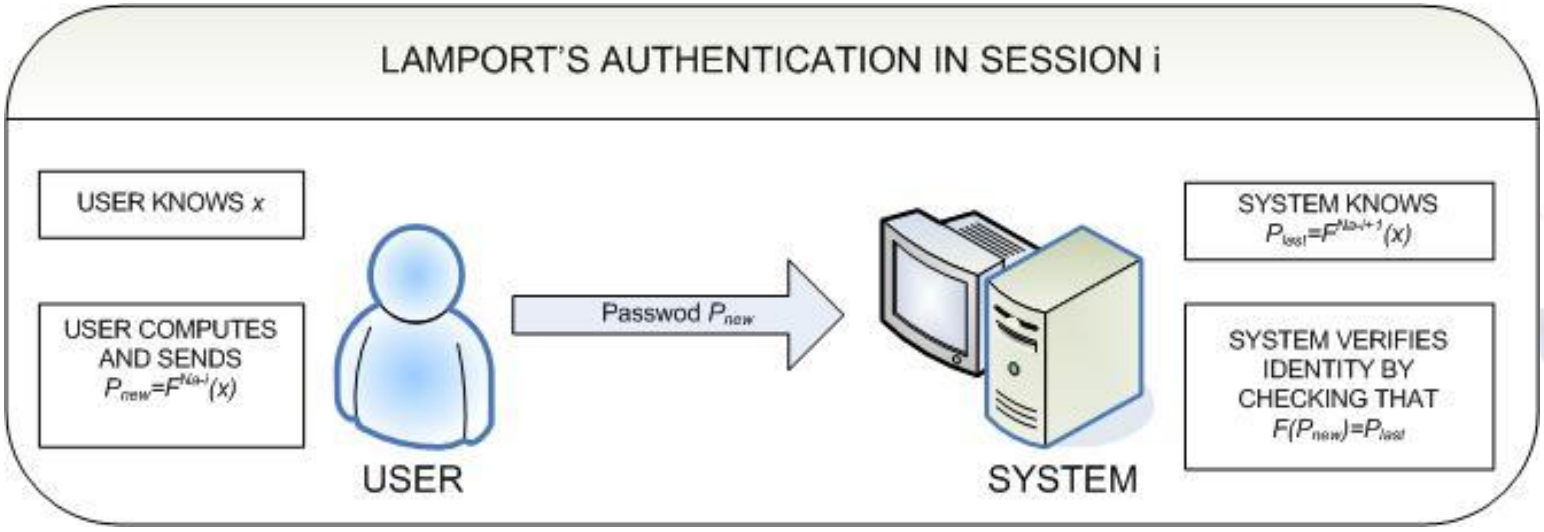
- **Principiu de functionare** al schemei Lamport:

- Fie *x* o valoare arbitrara aleasa de utilizator, *Na* numarul de autentificari necesare (de ex. 10000), *F* o functie one-way (de ex. o functie hash)
- Utilizatorul calculeaza de partea sa secventa $\{x, F(x), F(x), \ldots, F^{Na-1}(x), F^{Na}(x)\}$
- $F^{Na}(x)$ este trimisa sistemului intr-un stadiu off-line pentru a garanta autenticitatea acestei valori
- Cand utilizatorul se autentifica prima oara prezinta $F^{Na-1}(x)$ iar sistemul verifica daca $F(F^{Na-1}(x))=F^{Na}(x)$
- La modul general pentru a *i*-a autentificare parola este $F^{Na-i}(x)$

- Compozitia succesiva a functiei conduce la un lant one-way



- Fiecare element din lant joaca rol de parola



Schema Lamport in practica

- Exista o singura propunere de utilizare – sistemul S-Key [Haller, 1995]
- Poate fi spart printr-un atac de tip pre-play – un adversar poate impersona sistemul si poate captura parole inca nefolosite trimise de utilizator
- **Dezavantajul** schemei Lamport – **nu ofera autentificare mutuala**

Autentificare provocare-raspuns (challenge-response) (Autentificare puternica)

- **Principiu: Demonstreaza cunoasterea secretui** in baza caruia se demonstreaza identitatea **fara a dezvalui secretul**
- Se realizeaza prin calcularea unei valori numita raspuns pe baza altei valori numita challenge

1. A -> B : challenge

2. B -> A : response

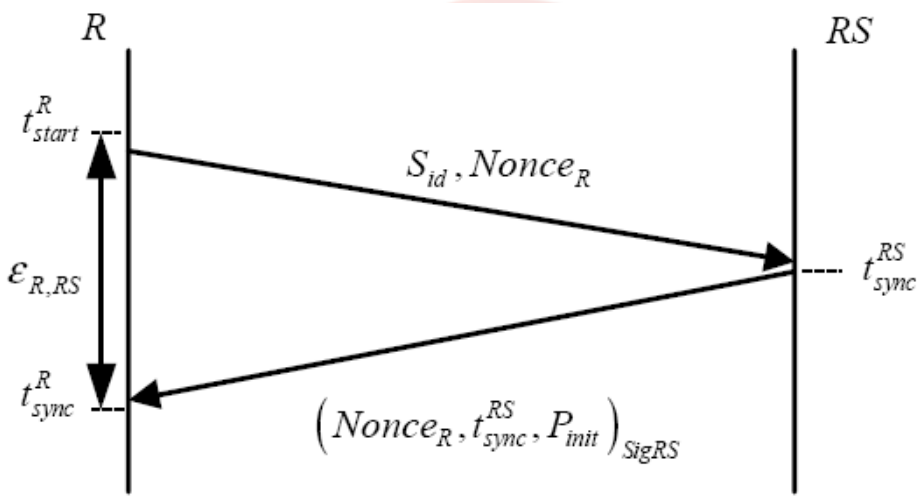
Parametrii varianti in timp

- Unul dintre obiectivele autentificarilor challenge-response este **eliminarea toatala a atacurilor de tip retransmisie replay**
- Parametrii varianti in timp sunt necesari pentru a preveni atacuri de tip replay prin garantarea unicitatii unei instante a unui protocol respectiv a cronologiei in unele cazuri (timeliness)
- Parametrii varianti in timp se numesc **Nonce** (a value used no more than once)
- Exista trei tipuri de parametrii varianti in timp

1) **Numere aleatoare** (cel mai comun utilizate) – numere generate aleator in fiecare instanta a protocolului (in general previn si atacuri de tip CPA si asigura unicitatea)
2) **Numere secventiale** – in general o valoare care se incremeneteaza la fiecare instanta a protocolului (utilizate pentru a detecta retransmisii)
3) **Amprente temporale** (timestamp) – asigura unicitate si timeline, previn atacuri de tip retransmisie – **necesita sincronizare temporala slaba**

Sincronizarea temporala slaba

- Element esential in multe protocoale de autentificare (in special in regim broadcast, vezi TESLA, curs 12)
- Principiul unei sincronizari temporale slabe: vezi desen, explicat la tabla



Challenge response cu functii criptografice simetrice

Identificare unilaterala bazata pe numere aleatoare

- **Principiu: utilizeaza un numar aleator ca si challenge**

$$A \rightarrow B: challenge$$

$$B \rightarrow A: response$$

Varianta 1 (poate fi atacata)

- A si B cunosc o cheie secreta k (r_A reprezinta un numar aleator generat de A)

$$A \rightarrow B: r_A$$
$$B \rightarrow A: E_k(r_A)$$

- A verifica identitatea lui B testand ca $E''_k(r_B) = E_k(r_B)$ ($E''_k(r_B)$ denota valoarea primita de A ca raspuns la challenge si care putea fi modificata de un adversar)

Atac prin reflexie asupra Variantei 1

$1.\ A \rightarrow Adv(B):\ r_A$

$1''.\ Adv(B) \rightarrow A:\ r_A$

$2''.\ A \rightarrow Adv(B):\ E_k(r_A)$

$2.\ Adv(B) \rightarrow A:\ E_k(r_A)$

- A a trimis un challenge catre adversarul Adv care pretinde ca este B iar intr-o noua instanta a protocolului Adv pretinde ca este B si ii cere lui A sa se autentifice folosind valoarea de autentificare a lui A ca autentificare pentru B in prima instanta a protocolului
- Rezultat A crede ca comunica cu B dar de fapt comunica cu Adv

Varianta 2 (Varianta 1 reparata in fata atacului prin reflexie)

- A si B cunosc o cheie secreta k (id_A reprezinta identitatea celui care a solicitat autentificarea)

$$A \rightarrow B: r_A$$
$$B \rightarrow A: E_k(r_A, id_A)$$

- A verifica identitatea lui B testand ca $E''_k(r_A) = E_k(r_A)$ si testand si ca identificatorul celui care a trimis challenge-ul este al sau

Atacul prin reflexie nu mai functioneaza asupra Variantei 2

$1.\ A \rightarrow Adv(B):\ r_A$

$1''.\ Adv(B) \rightarrow A:\ r_A$

$2''.\ A \rightarrow Adv(B):\ E_k(r_A, id_A)$

$2.\ Adv(B) \rightarrow A:\ E_k(r_A, id_A)$

- Atacul nu functioneaza pentru ca mesajul nu contine identificatorul lui A; adica $E_k(r_A, id_B) != E_k(r_A, id_A)$

Identificare unilaterala bazata pe time-stamp

- **Principiu: utilizeaza valoarea timpului** (cunoscuta de ambii participanti) **ca challenge**

$$A \rightarrow B: E_k(t_A, id_B)$$

- A verifica identitatea lui B testand ca $E''_k(t_A, id_B) = E_k(t_A, id_B)$ si testand ca timpul este timpul corect din sistem (mai exact se verifica ca timpul apartine unui interval de timp rezonabil, de ex. mesajul de autentificare provine din ultimele 5 secunde)

Identificare mutuala cu numere aleatoare

- **Observatie (legata de autentificarea mutuala):** Simpla dublare a unei autentificari unilaterale nu este in general o solutie deoarece nu exista nici o legatura intre cele doua instante ale protocolului de autentificare unilateral (sursa de atac)
- Autentificare mutuala

$$A \rightarrow B: r_A$$
$$B \rightarrow A: E_k(r_A, r_B, id_A)$$
$$A \rightarrow B: E_k(r_A, r_B)$$

- Se fac aceleasi verificari ca in cazurile precedente, in plus faptul ca valorile aleatoare sunt legate (fiind criptate impreuna) garanteaza caracterul mutual al autentificarii
- **Observatie (valabila pentru toate autentificarile challenge-response):** In orice autentificare challenge –response este recomandabil ca fiecare entitate ce executa o criptare cu o cheie secreta sa aplice functia de criptare nu doar pe valoarea de challenge ci sa mai adauge si o valoare aleatoare proprie (pentru a evita atacuri CPA). O alternativa este si utilizarea unei functii ireversibile cu cheie, de exemplu schimba $E_k(r_A, r_B, id_A)$ cu $MAC_k(r_A, r_B, id_A)$

Challenge-response cu tehnici asimetrice

- Aceleasi principiu al folosirii unei valori de challenge ca in cazul utilizarii tehnicilor simetrice
- Sistemele asimetrice pot fi utilizate in 2 abordari distincte:

a) **Decripteaza** o valoare criptata cu cheia s-a publica

b) **Semneaza digital** valoarea de challenge

Utilizarea criptarii asimetrice in challenge-response

- Varianta 1 (incorecta) – B isi demonstreaza identitatea decriptand mesaje la alegerea lui A

$$A \rightarrow B: E_{PbB}(r_A)$$
$$B \rightarrow A: r_A$$

- **Deficienta**: sursa perfecta de atacuri CCA2 asupra criptosistemului

Utilizarea criptarii asimetrice in challenge-response

- **Autentificare unilaterala prin utilizarea functiei de decriptare si a unei dovezi** (martor ,witness) al faptului ca A cunostea plaintextul: B isi demonstreaza identitatea decriptand mesajul trimis de A si verificand ca $h(r_A)$ corespunde hash-ului mesajului decriptat (deci A cunostea mesajul)
  - $A \rightarrow B: h(r_A), A, E_{PbB}(r_A, A)$
  - $B \rightarrow A: r_A$
- **Autentificare mutuala prin utilizarea functiei de decriptare**

$$A \rightarrow B: E_{PbB}(r_A, A)$$
$$B \rightarrow A: E_{PbA}(r_A, r_B)$$
$$A \rightarrow B: r_B$$

Utilizarea semnaturilor digitale in challenge-response

- Varianta 1 (incorecta)

$$A \rightarrow B: r_A$$
$$B \rightarrow A: Sig_B(r_A)$$

Exploatare asupra variantei 1

- A "pregateste" urmatoarea valoare de challenge pentru B
  - $r_A = h(\text{"Eu sunt B si doresc sa transfer 1.000.000 USD din contul meu in contul lui A"})$
- A trimite valoarea de challenge pentru B

$$A \rightarrow B: r_A$$

$$B \rightarrow A: Sig_B(r_A)$$

- B semneaza deoarece pentru el valoarea hash-ului r_A este un numar aleator fara nici o semnificatie
- A prezinta semnatura digitala a lui B $Sig_B(r_A)$ alaturi de mesaj $m=$*"Eu sunt B si doresc sa transfer 1.000.000 USD din contul meu in contul lui A"* autoritatii fiscale si obtine 1.000.000 USD
- **Morala**: nu semna mesaje alese integral de alt participant sau nu folosi aceeasi cheie si pentru semnaturi digitale si pentru autentificare (poti sa o folosesti pentru ambele, dar cu precautie)

Utilizarea semnaturilor digitale in challenge-response

- **Autentificare unilaterala cu time-stamp** (se verifica certificatului lui A $cert_A$ la autoritatea competenta (distribuitorul de certificate), validitatea semnaturii si faptul ca timpul la care a fost facuta semnatura este acceptabil)

$$A \rightarrow B: cert_A, t_A, B, Sig_A(t_A, B)$$

- **Autentificare unilaterala cu valori aleatoare** (se verifica certificatului lui A $cert_A$ la autoritatea competenta (distribuitorul de certificate), validitatea semnaturii si faptul ca a fost facuta pe valoarea de challenge)

$$A \rightarrow B: r_A$$
$$B \rightarrow A: cert_B, r_A, A, Sig_B(r_A, r_B, B)$$

- **Autentificare mutuala cu valori aleatoare** (aceleasi verificari ca in cazul precendent de partea ambilor participanti)

$$A \rightarrow B: r_A$$
$$B \rightarrow A: cert_B, r_B, A, Sig_B(r_A, r_B, B)$$
$$A \rightarrow B: cert_A, B, Sig_A(r_B, r_A, A)$$

Autentificare zero-knowledge

- **Principiu**: **demonstreaza o identitate fara a spune absolut nimic cu privire la secretul** in baza careia se face aceasta demonstratie
- Observatie: Protocoalele challenge-response au acest potential dezavantaj, un adversar poate selecta valori de challenge in mod strategic pentru a afla ceva despre secret
- Explicatie pentru copii - Pestera lui Quisquater et. al. Jean-Jacques Quisquater, Louis C. Guillou, Thomas A. Berson. **How to Explain Zero-Knowledge Protocols to Your Children**, *Advances in Cryptology - CRYPTO '89: Proceedings*, v.435, p.628-631, 1990.
- Structura generala a unui protocol z-k consta intr-o sesiune de 3 runde (deoarece argumentul fiecarei sesiuni este probabilistic devine necesara repetarea succesiva a acestei sesiuni cu noi valori)

$$A \rightarrow B: \ witness$$

$$B \rightarrow A: \ challenge$$

$$A \rightarrow B: \ response$$

Autentificarea z-k Fiat-Shamir

- **Obiectiv:** A isi demonstreaza identitatea catre B
- **Setarea cheii de autentificare:**

a) O parte de incredere T publica un modul RSA $n=pq$

b) A alege un numar s din Z_n si calculeaza $v=s^2 \bmod n$

c) v este cheia publica a lui A iar s este cea secreta (v se foloseste la verificare, s pentru generare raspunsului la challenge)

- **Descrierea unei runde:**

a) **A** alege un numar r (commitment) din Z_n si calculeaza si **trimite $x=r^2 \bmod n$ (witness)**

b) **B alege o valoare de challenge** $e=1$ sau $e=0$ si trimite e lui A

c) **A calculeaza raspunsul ca $y= r*s^e \bmod n$**

d) **B verifica daca $y^2 = xv^e \bmod n$**

- **Concluzia la final de runda: exista o probabilitate de exact ½ ca Adv(A) sa fii fraudat runda de autentificare** (explicatie pe slide-ul urmator)
- **Concluzia la finalul a k runde: exista o probabilitate de exact $(½)^k$ ca Adv(A) sa fii fraudat autentificarea**

Cum putea Adv(A) sa fraudeze o runda de autentificare cu probabilitate 1/2

- Cazul 1: **Adv(A)** poate selecta r, **calculeaza** $x=r^2/v$ si **raspunde cu** $y=r$ – acesta este un raspuns **corect pentru cazul in care** $e=1$ dar daca $e=0$ atunci Adv(A) trebuie sa calculeze radacina patrata a lui x, ceea ce este nefezabil
- Cazul 2: **Adv(A) poate raspunde corect in cazul in care** $e=0$ **daca calculeaza** $x=r^2 \bmod n$ dar in acest caz nu mai poate raspunde la $e=1$ pentru ca din nou trebuie sa poata calcula o radacina patrata

- Argument intuitiv pentru faptul ca este autentificare z-k: orice adversar poate simula valori corecte raspuns, dar nu poate poate sa raspunda in fata unei provocari real-time decat cu probabilitate $(½)^k$