SSI

1. Definitii laborator

- Adversar - O entitate (inclusiv un insider) care acționează rău intenționat pentru a compromite un sistem.
- Securitate - O condiție care rezultă din stabilirea și menținerea măsurilor de protecție care permit unei organizații/sistem să își îndeplinească misiunea sau funcțiile critice, în ciuda riscurilor reprezentate de amenințări.
- Risc - O măsură a gradului în care o entitate este amenințată de o eventuală circumstanță sau eveniment.
- Vulnerabilitate - Slăbiciune într-un sistem informațional, proceduri de securitate ale sistemului, controale interne sau implementare care ar fi putea fi exploatate sau declanșate de o sursă de amenințare.
- Securitate Cibernetica - Capacitatea de a proteja / apăra spațiul cibernetic de atacuri cibernetice.

- malware - software care este conceput special pentru a perturba, a deteriora sau a obține acces neautorizat la un sistem informatic
- virus - fragment de cod capabil să se copieze singur și să corupă / distrugă date dintr-un sistem, fără permisiunea sau cunoștința utilizatorului
- dropper - un tip de Trojan care a fost proiectat pentru a "instala" malware (virus, backdoor, etc.) pe un computer. Codul malware-ului poate fi conținut în interiorul dropper-ului.
- downloader - un tip de Troian care se instalează pe sistem și așteaptă până când o conexiune la internet devine disponibilă pentru a se conecta la un server sau site web de la distanță în scopul descărcării de programe suplimentare.
- trojan - un tip de cod sau software rău intenționat care pare legitim, dar poate prelua controlul asupra computerului
- spyware - software care permite unui utilizator să obțină informații ascunse despre activitățile computerului altcuiva prin transmiterea secretă a datelor de pe hard disk-ul lor.
- riskware - definește orice programe legitime care prezintă potențiale riscuri datorate vulnerabilităților de securitate, incompatibilității software sau încălcărilor legale.
- ransomware - un tip de software rău intenționat proiectat pentru a bloca accesul la un sistem informatic până când o sumă de bani este plătită.
- adware - software care afișează sau descarcă automat materiale publicitare, precum bannere sau pop-up-uri, atunci când un utilizator este online.
- worm - tip de malware care se răspândește prin copierea de sine de pe un computer pe altul.
- obfuscare - a face ceva dificil de înțeles. Codul de programare este adesea obfuscat pentru a proteja proprietatea intelectuală sau secretele comerciale și pentru a împiedica un atacator să facă reverse engineering unui program software.

- Criptologie - Știința care se ocupă de criptanaliză și criptografie.
- Criptografie - Disciplina care studiază principiile, mijloacele și metodele de transformare a datelor pentru a ascunde conținutul lor semantic, a preveni utilizarea lor neautorizată sau a preveni modificarea lor nedetectată.
- Criptanaliza - Încercarea de a înfrânge protecția criptografice fără o cunoaștere inițială a cheii utilizate în furnizarea protecției.
- Confidentialitate - Asigurarea că informațiile nu sunt dezvăluite entităților neautorizate.
- Integritate - Protejarea împotriva modificării sau distrugerii necorespunzătoare a informațiilor.
- Disponibilitate - Asigurarea accesului și utilizării informațiilor în timp util și fiabil.

Pentru fiecare dintre afirmațiile de mai jos, indicați proprietatea (proprietățile) CIA la care se face referire:

- Salariile angajaților nu trebuie făcute publice - C
- Biroul casierie trebuie să aibă acces la salariile angajaților (pentru a realiza plățile) - D
- Un angajat nu își poate modifica singur suma primită ca salariu pe luna în curs - I
- Un angajat nu ar trebui să afle cât câștiga un coleg fără acordul acestuia (ex. să îi spună direct) - C
- Biroul casierie trebuie să aibă certitudinea că suma pe care o înmânează angajatului de plată este cea corectă - I

Răspundeți cu adevărat sau fals:

- Un adversar care are la dispoziție un timp infinit pentru criptanaliza unui sistem este un adversar PPT - **F**
- Un adversar PPT are dreptul de a „ghici” cheia - **A**
- Un adversar PPT are la dispoziție algoritmi exponențiali în timp - **F**

Care dintre următoarele funcții sunt neglijabile în parametrul de securitate n, având în vedere un adversar PPT?

- $f(x)$=2 - neneglijabila
- $f(x)$=1/2000 - neneglijabila
- $f(x)$=1/n^{2000} - neneglijabila
- $f(x)$=1/2^n - neglijabila
- $f(x)$=$f1(x)$+$f2(x)$, unde $f1(x)$ și $f2(x)$ sunt neglijabile - neglijabila
- $f(x)$=$f1(x)$+$f2(x)$, unde $f1(x)$ este neglijabilă și $f2(x)$ este neneglijabilă - neneglijabila

Se consideră un sistem criptografic care folosește o cheie de criptare pe 512 biți.

- Câte chei posibile distincte există? - 2^512 (Cheie de criptare pe 512 biti.)

- Cât timp îi va lua unui adversar găsirea cheii corecte (cazul cel mai nefavorabil) dacă are la dispoziție un calculator care testează 2^{30} chei pe secundă? - 2^{482} (2^{30} chei pe secundă => timp $2^{(512-30)} = 2^{482}$.
- Considerați că este un atac eficient? - Nu este un atac eficient datorită timpului.

Ce impact are folosirea cheii în cazul OTP pentru o altă criptare?

Dacă un atacator are acces la un mesaj criptat și la mesajul clar, atunci poate afla cheia și respectiv toate mesajele criptate cu aceeași cheie.

- Inginerie socială - O încercare de a păcăli pe cineva să dezvăluie informații (de exemplu, o parolă) care pot fi folosite pentru a ataca sisteme sau rețele.
- Phishing - O tehnică pentru încercarea de a achiziționa date sensibile, cum ar fi numerele de cont bancar, printr-o solicitare frauduloasă prin e-mail sau pe un site web, în care făptuitorul se maschează ca o afacere legitimă sau o persoană de încredere.
- Whaling - Un tip specific de phishing care vizează membrii de rang înalt ai organizațiilor.
- Pharming - Utilizarea mijloacelor tehnice pentru a redirecționa utilizatorii către accesarea unui site Web fals, mascat drept unul legitim și divulgarea informațiilor personale.
- Spear phishing - Un termen colocvial care poate fi folosit pentru a descrie orice atac de phishing foarte vizat.
- Spoofing - Falsificarea adresei de trimitere a unei transmisii pentru a obține intrarea ilegală într-un sistem securizat.

Răspundeți cu adevărat sau fals pentru fiecare dintre următoarele afirmații. Căutați online informații despre funcțiile hash menționate.

- Amestecarea ingredientelor pentru realizarea unei prăjituri poate fi considerată one-way function - **A**
- Funcția hash MD5 este considerată sigură la coliziuni. - **F**
- SHA256 este o funcție hash cu output pe 256 biți - **A**
- Este corect să afirmăm că „o funcție hash criptează” - **F**

- O funcție hash folosită pentru stocarea parolelor trebuie să fie rapidă (i.e., să se calculeze rapid H(x) pentru x dat) - **F**

2. Triada CIA

- **Confidentialitate**: păstrarea secretului informației, accesul la informația sensibilă fiind disponibilă doar persoanelor autorizate.
- **Integritate** (a datelor): eliminarea posibilității de modificare (schimbare, inserare, stergere) neautorizată a informației.
- **Disponibilitate**: permiterea entitatilor autorizate sa acceseze în timp util si fiabil informatia.

- **Autentificare**: identifica o entitate sau atestă sursa datelor.
- **Non-repudiere**: previne negarea unor evenimente anterioare.

Def. **Criptografia** este studiul tehnicilor matematice relationate cu aspecte ale securității informației precum confidențialitatea, integritatea datelor, autentificarea entitatilor sau a originii datelor.

Definitie

Un sistem de criptare simetric definit peste $(\mathcal{K}, \mathcal{M}, \mathcal{C})$*, cu:*

- $\mathcal{K}$ = *spațiul cheilor*
- $\mathcal{M}$ = *spațiul textelor clare (mesaje)*
- $\mathcal{C}$ = *spațiul textelor criptate*

este un dublet $(\mathrm{Enc}, \mathrm{Dec})$*, unde:*

1. $\mathrm{Enc} : \mathcal{K} \times \mathcal{M} \to \mathcal{C}$
2. $\mathrm{Dec} : \mathcal{K} \times \mathcal{C} \to \mathcal{M}$

a.î. $\forall m \in \mathcal{M}, k \in \mathcal{K} : Dec_k(Enc_k(m)) = m.$

Terminologie

- Mesajul in forma originară se numeste **text clar**.

- Expeditorul rescrie mesajul folosind un sistem de criptare, adică îl criptează și obține un **text criptat**.
- Destinatarul îl **decripteaza** cunoscând metoda folosită pentru **criptare**.
- Procesul de determinare a cheii aferente unui sistem de criptare, cunoscând doar textul criptat (eventual si alte informatii auxiliare) se numeste **criptanaliza**.
- Decriptarea si criptanaliza au acelasi scop: găsirea textului clar; diferenta consta în faptul ca la criptanaliza nu se cunoaște cheia de decriptare.

Scenarii de atac

- **Atac cu text criptat**: Atacatorul știe doar textul criptat - poate încerca un atac prin forța brută prin care se parcurg toate cheile pană se găsește cea corecta.
- **Atac cu text clar**: Atacatorul cunoaște una sau mai multe perechi (text clar, text criptat).
- **Atac cu text clar ales**: Atacatorul poate obține criptarea unor texte clare alese de el.
- **Atac cu text criptat ales**: Atacatorul are posibilitatea sa obtina decriptarea unor texte criptate alese de el.

Estimarea timpului de succes pentru un atac de tip *forță brută* asupra unui sistem de criptare *simetric*:

Lungime cheie (biți)	Securitate estimată
56 - 64	termen scurt (ore sau zile)
112 - 128	termen lung (în absența calculatoarelor cuantice)
256	termen lung (în prezenta calculatoarelor cuantice, folosind algoritmii cunoscuți)

Principiul lui Kerckhoffs. *Sistemul nu trebuie să fie secret, poate să cadă ușor în mâinile adversarului (i.e. securitatea unui sistem de criptare nu constă decât în menținerea secretă a cheii).*

- este mai ușor de păstrat o *cheie secretă* decât un *algoritm secret*;
- este mai ușor de schimbat o *cheie compromisă* decât un *algoritm compromis*;
- permite standardizarea algoritmilor de criptare.

3. Sisteme de criptare istorice

Cifruri de permutare / transpozitie

Cifru de permutare = rearanjarea literelor în textul clar pentru a obține textul criptat

	→			
	M	A	R	T
↓	E	J	I	A
	S	C	P	T

Text clar: mesaj criptat
Cheia: $k = 3$
Text criptat: MARTEJIASCPT

Cifru de substituție = inlocuirea unui caracter (set de caractere) cu un alt caracter (set de caractere). Pot fi monoalfabetice (un caracter e întotdeauna înlocuit cu același caracter) sau polialfabetice (un caracter este locuit cu caractere diferite)

Cifruri de substitutie monoalfabetice

Cifrul lui Cezar

a	b	c	d	e	f	g	h	i	j	k	l	m
D	E	F	G	H	I	J	K	L	M	N	O	P

n	o	p	q	r	s	t	u	v	w	x	y	z
Q	R	S	T	U	V	W	X	Y	Z	A	B	C

Text clar: mesaj criptat
Text criptat: PHVDM FULSWDW

- $\mathcal{K} = \{0, 1, \ldots, 25\}$
- $\mathcal{M} = \{a, b, \ldots, z\}^*$
- $\mathcal{C} = \{A, B, \ldots, Z\}^*$
- $\mathrm{Enc} : \mathcal{K} \times \mathcal{M} \to \mathcal{C}$

$$Enc_k(m) = m + k \pmod{26}$$

- $\mathrm{Dec} : \mathcal{K} \times \mathcal{C} \to \mathcal{M}$

$$Dec_k(c) = c - k \pmod{26}$$

Principiul cheilor suficiente

Principiul cheilor suficiente: *O schemă sigură de criptare trebuie să aibă un spațiu al cheilor suficient de mare a.î. să nu fie vulnerabilă la căutarea exhaustivă.*

Substitutia simpla

Asemănătoare cifrului lui Cezar, doar ca nu se pastreaza permutarea cu o litera la o distanta fixa, poate fi oricare

|K| = 26! Deci atacul brute force devine mai dificil de efectuat

Sistemul afin

Structura de bază a unui sistem afine poate fi exprimată printr-o funcție matematică de criptare și o funcție matematică de decriptare. O astfel de funcție afina pentru criptare (E) și decriptare (D) ar putea arăta astfel:

Criptare (E):	Decriptare (D):
$E(x)=(ax+b) \bmod m$	$D(y)=a^{-1}(y-b) \bmod m$

În aceste formule:

- x reprezintă textul clar,
- y reprezintă textul criptat,
- a și b sunt constante, (a întreg pozitiv coprim cu m, b poate fi orice număr întreg)
- m este dimensiunea alfabetului (numărul de caractere posibile).

Analiza de frecventa

Determinarea corespondentei intre alfabetul clar si alfabetul criptat pe baza frecvenței de apariție a literelor în text, cunoscând distribuția literelor în limba textului clar (se cunoaște limba textului clar, iar lungimea textului permite analiza de frecvență)

Cifruri de substitutie polialfabetice -> sistemul Playfair, Vigenere

Aceste sisteme sunt total nesigure.

4. Securitate perfecta (OTP)

O schema de criptare peste un spațiu al mesajelor M este perfect sigură dacă pentru orice probabilitate de distribuție peste M, pentru orice mesaj m din M și orice text criptat c pentru care:

- Pr[C=c] > 0 are loc egalitatea: Pr[M=m | C=c] = Pr[M=m].
- Pr[M=m] - probabilitatea dinainte ca Alice să aleagă mesajul m
- Pr[M=m | C=c] - probabilitatea după ce a Alice a ales mesajul m, chiar dacă textul criptat a fost văzut.

Securitatea perfectă – dacă Oscar afla textul criptat nu are niciun fel de informație în plus decat dacă nu l-ar fi aflat.

O schema de criptare (Enc, Dec) este perfect sigură dacă pentru orice mesaje m0, m1 din M cu |m0| = |m1| si orice c din C are loc: Pr[Enc k(m0) = c] = Pr[Enc k(m1) = c] unde k este o cheie aleasă uniform.

Fiind dat un text criptat este imposibil de ghicit dacă textul clar este m0 sau m1.

Cel mai puternic adversar nu poate deduce nimic despre textul clar dat fiind textul criptat.

One Time Pad (OTP) = Cifrul Vernam

Algoritm:

Fie I > 0, M=C=K = $\{0,1\}^I$

Cheia k se alege cu distributie uniforma din spatiul cheilor k

Enc: data o cheie k din $\{0,1\}^I$ si un mesaj m din $\{0,1\}^I$, intoarce c = k xor m

Dec: data o cheie k din $\{0,1\}^I$ si un mesaj criptat c din $\{0,1\}^I$, intoarce m = k xor c

- avantaj = criptare/decriptare rapida
- dezavantaj = cheie de lungimea mesajului (foarte lunga)
- acelasi text criptat poate sa provina din orice text clar cu o cheie potrivita

Schema de criptare OTP este perfect sigură, dar este nepractica.

Nu exista nicio schema de criptare (Enc, Dec) perfect sigură în care mesajele au lungimea n biți, iar cheile au cel mult lungimea n-1 biti.

Dacă se reutilizeaza cheia, atacatorul poate face XOR între cele două mesaje criptate. Deoarece A XOR B XOR A = B, atacatorul obține textul clar pentru al doilea mesaj.

5. Securitate computationala

Sunt de interes mai mare schemele care practic nu pot fi sparte deși nu beneficiază de securitate perfectă (sunt sigure în fața adversarilor eficienti care executa atacul într-un interval de timp realizabil, iar succesul are o probabilitate foarte mica)

O schema este sigură dacă orice adversar care dispune de timp polinomial in n (parametrul de securitate) efectuează un atac cu succes numai cu o probabilitate neglijabilă.

În practică x este scalar:

x ne-neglijabil daca x >= ½^30

x neglijabil daca x <= ½^128

În teorie x este funcție și p(n) functie polinomiala in n:

- x ne-neglijabila in n daca exista p(n) astfel incat x(n) >= 1/p(n)
- gasesti 1 / functie polinomiala mai mica decat x(n), de ex 1/n^2000 este ne-neglijabila

x neglijabilă în n dacă pentru orice p(n) exista nd astfel incat orice n >= nd are loc relatia x(n) < 1/p(n)

→ 1 / orice functie polinomiala mai mare decat x(n), de ex $(\frac{1}{2})^n$ este neglijabila

! Orice functie polinomiala este ne-neglijabila.

Astfel, sistemul ramane sigur pentru un adversar PPT (Probabilistic polinomial in timp)

6. PseudoRandom Generator (PRG)

- Acesta este un algoritm **determinist** care primește o "sămânță" relativ scurtă s (*seed*) și generează o secvență *pseudoaleatoare* de biți;
- Notăm $|s| = n$, $|PRG(s)| = l(n)$
- PRG prezintă interes dacă:

$$l(n) \geq n$$

(altfel NU "generează aleatorism")

Deci **PRG**, poate fi ușor predictibil în funcție de cum este conceput.

Fie I un polinom si G un algoritm polinomial determinist si oricare s din {0,1}^n, G generează o secvență de lungime I(n).

Dacă G are expansiune (I(n) >= n) si pseudoaleatorism, atunci G este generator de numere pseudoaleatoare (PRG).

I = factorul de expansiune al lui G.

Sisteme fluide

Un sistem de criptare (Enc, Dec) definit peste (K, M, C) se numeste sistem de criptare fluid daca:

- Enc : K x M -> C si c = Enc k(m) = G(k) xor m
- Dec : K x C -> M si m = = Dec k(c) = G(k) xor c

Unde G este un generator de numere pseudoaleatoare cu factorul de expansiune I, k din {0, 1}^n, m din {0,1}^I(n)

Dacă G este PRG atunci sistemul fluid definit anterior este un sistem de criptare simetric de lungime fixă computational sigur pentru un atacator pasiv care poate intercepta un mesaj.

Moduri de utilizare:

Un sistem de criptare fluid in aceasta varianta = determinist (unui text clar îi corespunde întotdeauna același mesaj criptat), astfel devine nesigur pentru criptarea mai multor mesaje cu aceeași cheie. PRG va necesita 2 intrari: cheia k și un vector de inițializare IV.

În practica se folosește în 2 moduri:

- Sincronizat
    - Mesajele sunt criptate în mod succesiv, necesita păstrarea stării, mesajele succesive pot fi percepute ca un singur mesaj clar lung (obtinut prin concatenarea mesajelor succesive) și se preteaza pt o singura sesiune de comunicatie
- Nesicronizat
    - Mesajele sunt criptate independent, nu necesita păstrarea stării, valorile IV sunt alese uniform și fac parte din mesajul criptat
    - Trebuie sa satisfaca cel puțin
        - G(s, IV) este o secventa pseudoaleatoare chiar dacă IV este public (securitatea lui G consta in securitatea lui s)
        - Dacă IV1 și IV2 sunt valori uniform aleatoare, atunci G(s, IV1) si G(s, IV2) sunt indistinctibile.
    - Exemple: RC4, WEP, SEAL, A5/1

7. Linear Feedback Shift Register (LFSR)

LFSR este un tip de dispozitiv de stocare a datelor în care biții se deplaseaza in serie printr-un set de celule, iar starea depinde de o operatie de feedback liniara. LFSR poate fi folosit în generatori de numere pseudo aleatoare.

Un LFSR poate fi definit prin relatia de recurenta, ce arata cum bitii din starea actuala influenteaza urmatorul bit de iesire si cum se deplaseaza starea registrului.

LFSRn (t) = xor(ci LFSRi (t)) pentru i de la n-1 la 0
LFSRn (t) = (cn-1 LFSRn-1 (t)) xor (cn-2 LFSRn-2 (t)) xor ... xor (c0 LFSR0 (t))

Determinarea primilor biti de output (exemplu):

Fie LFSR cu 4 biti si relatia de recurenta: LFSR(t) = bitul de pe poziția 0 xor bitul de pe poziția 2, adică următorul bit de ieșire este dat de xor dintre ultimul bit și bitul de la poziția a3a

! Scriu in paranteze biții care sunt shiftati

Fie starea initiala LFSR = 1101

1. LFSR(t) = 1110 (1) (apare 1 pe poziția 0 de la 1 (poz 0 initial) xor 0 (poz 2 initial)
2. LFSR(t) = 0111 (01)
3. LFSR(t) = 1011 (101)
4. LFSR(t) = 0101 (1101)
5. LFSR(t) = 0010 (11101)
6. LFSR(t) = 1001 (011101)
7. LFSR(t) = 1100 (1011101)
8. LFSR(t) = 1110 (01011101) PERIOADA 7 (s-a repetat pasul 1)
9. LFSR(t) = 0111 (001011101)
10. LFSR(t) = 1011 (1001011101)

Perioada maximă a unui LFSR cu n biți este de 2^n – 1.

Sistemele de criptare ***simetrice*** pot fi:

- fluide (cripteaza bit cu bit, iar criptarea unui bit din textul clar este independenta de restul criptarilor de biti, folosesc PRG – cifruri, LFSR, PRG, OTP)
- bloc (cripteaza cate n biti simultan, iar criptarea unui bit este dependentă de biții din textul clar care apartin aceluiasi bloc, folosesc PRP).

Sistemele fluide sunt mai puțin sigure decat cele bloc.

Permutare pseudoaleatoare (PseudoRandom Permutation) PRP

PRP = functie bijectiva și determinstă care pentru o cheie fixă produce la ieșire o permutare a intrării indistinctbila fata de o permutare aleatoare. Functia aceasta si inversa ei sunt eficient calculabile.

O permutare pseudoaleatoare definită peste (K, X) e o functie bijectiva F : X x K -> X care satisface:

1. Eficienta (exista algoritmi determinisi PPT care calculeaza F și F^(-1)
2. Pseudoaleatorism

Funcție pseudoaleatoare (PseudoRandom Function) PRF

- generalizarea noțiunii de permutare pseudoaleatoare
- este o functie cu cheie care este indistinctibila fata de o functie aleatoare cu același domeniu și mulțime de valori și satisface aceleași principii ca PRP

Sisteme bloc

Daca lungimea mesajului clar este mai mica decat dimensiunea unui bloc atunci se completeaza cu biți. Dacă lungimea mesajului clar este mai mare decat lungimea unui bloc se utilizează un mod de operare (ECB, CBC, OFB, CTR).

8. Criptografia simetrica si criptografía asimetrica

Criptografia se imparte in criptografie simetrica si criptografie asimetrica.
Criptografia simetrica va contine sisteme de securitate fluide și sisteme de securitate bloc și folosesc aceeași cheie pentru criptare și pentru decriptare.

Limitarile criptografiei simetrice

1. Distribuirea cheilor (pentru o transmitere sigură a cheilor secrete este necesar un canal de comunicatie sigur ceea ce este nepractic pentru organizații mari cu mai multe sedii în mai multe țări de exemplu)
2. Stocarea secretă a cheilor (pentru o companie cu n angajați sunt necesare N(N-1)/2 chei pentru o comunicare criptata la care se adaugă numărul cheilor pentru accesul la resurse, iar cu cat sunt mai multe chei, cu atât sunt mai dificil de stocat ceea ce diminuează securitatea sistemului) (pentru număr relativ redus de chei se poate folosi smartcard)
3. Medii de comunicare deschise
4. Imposibilitatea non-repudierii (o cheie simetrică este deținută de cel puțin 2 părți și este imposibil de demonstrat ca un MAC a fost produs de una dintre cele 2 parti comunicante, deci nu se poate utiliza autentificarea simetrică pentru a garanta sursa unui mesaj)

Criptografia asimetrica folosește chei diferite pentru criptare si pentru decriptare.
Un sistem de criptare asimetric definit peste (K, M, C) cu K = K pk x K sk (pk e cheia publica, iar sk este cheia secreta) spațiul cheilor, M spațiul mesajelor, C spatiul textelor critpate este un dublet (Enc, Dec) unde:

1. Enc : K pk x M -> C
2. Dec : K sk x C -> C

Astfel încât pentru orice m din M și (pk, sk) din K are loc Dec sk (Enc pk (m)) = m.

Cheia publica este larg raspundita pentru a asigura posibilitatea de criptare oricui dorește sa transmită un mesaj către entitatea care îi corespunde și se foloseste la criptare, iar cheia secretă se foloseste la decriptare și este privată nestiind decât entitatea căreia îi corespunde.

Criptare simetrica Criptare asimetrica

Necesita secretizarea întregii chei	Necesita secretizarea unei jumătăți de chei
Folosește aceeași cheie pt criptare și decriptare	Folosește chei distincte pentru criptare și decriptare
Rolurile emițătorului și a interpretorului pot fi schimbate	Rolurile emițătorului și receptorului NU pot fi schimbate
Pentru ca un utilizator sa primească mesaje criptate de la mai multi emitatori trebuie sa partajeze cu fiecare cate o cheie	O pereche de chei asimetrice permite oricui sa transmita informație criptată către entitatea căreia îi corespunde

Criptarea asimetrica

- Avantaje
    - Numar mic de chei
    - Simplifica problema distribuirii cheilor
    - Fiecare participante trebuie sa stocheze o singura cheie secretă de lungă durata
    - Permite comunicarea sigură pe canale publice
    - Rezolva problema mediilor de comunicare deschise
- Dezavantaje
    - Este mai lenta decat cea simetrică
    - Compromiterea cheii private conduce la compromiterea tuturor mesajelor primite indiferent de sursa
    - Necesita verificarea autenticității cheii publice

9. Sisteme de criptare simetrice

Moduri de operare bloc

ECB (Electronic Code Block)

- paralelizabil
- determinist (deci nesigur)
- F trebuie sa fie inversabila
- un adversar pasiv poate detecta repetarea unui bloc de text clar pentru ca se repeta blocul criptat corespunzător. ECB nu trebuie utilizat in practica

CBC (Cipher Block Chaining)

- IV (vectorul de inițializare) este ales aleator la criptare si se transmite in clar pentru decriptare.
- F trebuie sa fie inversabila
- este secvential (dezavantaj major dacă se poate utiliza procesarea paralelă)

CTR (counter)

- genereaza o secventa pseudoaleatoare care se xor-eaza mesajului clar
- ctr este ales aleator la criptare si se transmite in clar pt decriptare
- F nu e nevoie sa fie inversabila
- paralelizabil
- exemple DES (Data Encryption Standard) si AES (Advanced Encryption Standard)

DES

- rețea de tip Feistel cu 16 runde si o cheie pe 56 biti
- derivarea cheii obtine o sub cheie de runda ki pt fiecare runda pornind de la cheia master k (fiecare sub cheie ki reprezinta permutarea a 48 de biti din cheia master)
- fiecare cheie ki este cunoscută, doar cheia master este secreta
- pentru fiecare runda mesajul de 64 de biti este impartit in 2 parti de cate 32 de biti. Partea din stanga se xor-eaza cu rezultatul functiei aplicate pe partea dreapta si se muta in dreapta, iar partea din dreapta se muta în partea stanga. (desen pag 238)
- securitatea sistemului DES poate fi compromisă prin atacuri de tip brute force (+ lungimea cheii este prea mica)
- triplu DES (3DES) se bazează pe tripla invocare a lui DES folosind 2 sau 3 chei

Atacul meet-in-the-middle

Daca se cunoaste o pereche de text clar / text criptat (x, y) cu y = F k2(F k1(x)):

1. Pentru fiecare k1 din {0,1}^n calculeaza z = F k1(x) si pastreaza (z, k1)
2. Pentru fiecare k2 din {0,1}^n calculeaza z = (F k2)^(-1) si pastreaza (z, k2)
3. Verifica daca exista perechi (z, k1) si (z, k2) care coincid pe prima componenta
4. Atunci valorile k1 si k2 satisfac F k1(x) = F k2)^(-1)(y) adica y = F'k1, k2 (x)

Complexitatea acestui atac este 2^n.

AES

- este o retea de substitutie – permutare pe 128 biti care poate folosi chei de 128, 192 sau 256 de biti
- pt lungime de 128 – 10 runde, 192 – 12 runde, 256 – 14 runde
- folosește o matrice de 4x4 octeți = stare
- starea este modificata prin 4 tipuri de operații
- ultima runda => textul criptat
- cele 4 operatii sunt:
    - AddRoundKey: XOR cu cheia de runda
    - SubBytes: fiecare octet e inlocuit de un alt octet conform tabelei de substitutie S-box (sbox e o permutare), iar tabela e unica pt AES
    - ShiftRows
    - MixColumns

Ultimele doua reprezinta amestecarea bitilor folosind o transformare liniara pe biti.

→ pentru ***AES*** nu exista un atac mai eficient decat cautarea exhaustiva pt AES cu numar complet de runde.

10. Message Authentication Code (MAC)

Coduri de autentificare a mesajelor MAC (Message Authentication Code)

- au scopul de a împiedica un adversar sa modifice un mesaj trimis fără ca partile care comunica să nu detecteze modificarea
- cand se transmite un mesaj m de la A la B, A calculează mai întâi un tag t pe baza mesajului m și a cheii k si trimite (m, t)
- tag-ul e calculat folosind un algoritm de generare a tag-urilor numit Mac
- la primirea cheii (m, t) B verifica dacă tag-ul este valid în raport cu cheia k folosind un algoritm de verificare Vrfy

Un cod de autentificare a mesajelor MAC definit peste (K, M, T) este format dintr-o pereche de algoritmi polinomiali (Mac, Vrfy) unde:

1. Mac : K x M -> T este algoritmul de generare al tag-urilor
2. Vrfy : K x M x T -> {0,1} este algoritmul de verificare ce întoarce un bit b (1 valid, o altfel)

Astfel incat orice m din M și orice k din K are loc relația Vrfy k (m, Mac k(m)) = 1

Securitate MAC

- presupunem ca adversarul acționează în timp polinomial și are acces la mesajele trimise între părți împreuna cu tagurile aferente, deci este un adversar activ
- adversarul are acces la un oracol care returnează un tag pentru orice mesaj
- securitatea este impactata dacă adversarul este capabil să producă un mesaj m împreuna cu un tag t astfel incat
    - t este un tag valid pentru mesajul m: Vrfy k(m, t) = 1
    - Adversarul nu a solicitat anterior un tag pentru mesajul m

Un cod de autentificare al mesajelor pi = (Mac, Vrfy) este sigur dacă nu poate fi falsificat printr-un atac cu mesaj ales.

- nu oferă niciun fel de protectie pentru atacurile prin replicare efectuate de părțile comunicante deoarece definitia să nu incorporeaza nicio noțiune de stare în algoritmul de verificare
- totusi, folosind secvențe de numere sau ștampila de timp, mac-ul poate fi protejat
- pentru secvențe de numere, fiecare mesaj m are asignat un număr i, iar tagul este calculat pe mesajul i || m astfel orice nouă replicare a lui m trebuie sa construiasca un tag pentru mesajul i' || m, unde i nu a mai fost folosit niciodată. (dezavantajul este ca trebuie stocată o lista cu numerele folosite anterior).
- pentru ștampila de timp se verifica dacă ștampila inclusă în mesaj se afla la un interval de timp acceptabil, presupunand ca părțile au ceasurile sincronizate
- dacă un atac prin replicare este suficient de rapid, el se poate desfășura cu succes chiar și în aceste condiții
- pentru a construi mac-uri sigure se folosesc funcții pseudoaleatoare (PRF)

MAC-uri pentru mesaje de lungime variabila

Sunt folosite în practica și se construiesc pornind de la MAC-urile de lungime fixă folosind următoarele abordări: xorarea pe toate blocurile cu autentificarea rezultatului (nesigur deoarece

atacatorul poate obține un tag valid pentru un mesaj nou modificand mesajul original), autentificarea separată pentru fiecare bloc (nesigur, atacatorul poate schimba ordinea blocurilor obtinand un tag valid pentru un mesaj nou), autentificarea separată pt fiecare bloc folosind o secvență de numere (nesigur, un adversar poate scoate blocuri de la finalul mesajului și obține un tag valid pentru un mesaj nou de lungime mai mica)

O soluție sigură ar fi adăugarea de informație suplimentară în fiecare bloc în afara numărului de secvență, dar este ineficienta.

CBC-MAC

Mac sigur utilizând cifruri bloc. Are o construcție similară cu modul CBC pentru criptare. Folosind CBC-MAC pentru un tag aferent unui mesaj de lungime I x n se aplica sistemul bloc doar de I ori. Folosit in industria bancara, WPA2

HMAC

Folosit pt protocoalele de internet: SSL, SSH. Mac clasic + hash

CPA (Chosen Plaintext Attack)

Adversarul pentru un text află criptarea.

Un adversar nu poate determina care text clar a fost criptat cu o probabilitate semnificativ mai mare decat dacă ar fi ghicit, chiar dacă are acces la oracolul de criptare. (nu știe a cui criptare din doua texte de exemplu)

CCA (Chosen Ciphertext Attack)

Adversarul pentru o criptare afla decriptarea și pentru un text criptarea

Dacă un atacator poate obține decriptări pentru texte cifrate alese, nu va putea să obțină informații utile despre cheia secretă sau să efectueze alte atacuri periculoase.

CCA sigur presupune necunoasterea funcției de criptare a mesajului

Pentru criptarea unui mesaj m, A procedeaza astfel:

- Criptează m folosind schema CPA sigură rezultat textul criptat c
- Calculeaza un tag MAC t pe textul criptat c
- Rezultatul final al criptării este (c, t)

Pentru un text criptat (c, t), Bob verifica validitatea tag-ului înainte de a decripta

Un text criptat (c, t) este valid daca t este un tag valid pt c.

Confidentialitatea provine din scheme de criptare, integritatea din utilizarea unui MAC.
Exista 3 scheme de combinare a criptării si autentificarii mesajelor:

1. Criptare si autentificare -> criptare si autentificare (nu este sigura, un MAC nu implica confidentialitate)
2. Autentificare apoi criptare -> nu este sigura
3. Autentificare apoi criptare -> sigura

GCD (Euclid Efficient)

```
def gcd(a,b):
    if (b == 0):
        return a
   return gcd(b, a%b)
```

Simbolul Legendre

a/p (unde a este număr întreg iar p este număr prim)

- 0 daca a mod p == 0
- 1 daca exista un intreg x ai x^2 mod p == a
- -1 altfel

Simbolul Jacobi

a/n unde a este un număr întreg iar n este un număr întreg pozitiv

(a/n) = produs de la i = 1 pana la k (a/pi)^ei unde n este scris ca un produs de puteri de numere prime distincte astfel n=p1^e1 * p2^e2 * ... * pn ^en

11. Notiuni preliminare

…

12. Sistemul de criptare RSA

- cel mai cunoscut si utilizat algoritm cu cheie publica
- problema RSA se bazează pe dificultatea factorizării numerelor mari: N=p*q, p si q prime

Fie Z^*_N un grup de ordin teta(N) = (p-1)(q-1)

Problema RSA

Fiind dat N, un intreg e > 0 care satisface (e, teta(n)) = 1 si un element y din Z^*_N se cere sa se gaseasca x ai x^e = y mod N.

Dacă se cunoaște factorizarea lui N, atunci teta(N) este ușor de calculat și problema RSA este ușor de rezolvat, dacă nu se cunoaște teta(N) problema RSA este dificila.

Probleme de securitate a RSA:

- Fiind determinist, Textbook RSA nu este CPA sigur si nici CCA sigur
- Coeficientul de criptare mic nu este o alegere pentru RSA deoarece orice mesaj m<N^1/e nu folosește reducerea modulară la criptare c = m^e mod N = m^e. Fiind dat c, se determina imediat m ca m = c^1/e mod N
- Atac cu text criptat ales.
- Utilizarea multipla a modulului. Nu trebuie utilizate mai multe perechi de chei care folosesc același modul

RSA

- algoritm de criptare cu cheie publica ce utilizeaza doua chei diferite pentru criptare si decriptare: o cheie publica si o cheie privata
- generarea cheilor
    - Pentru a cripta/decripta date, RSA trebuie sa genereze o cheie privata si una publica
    - Se aleg doua numere prime mari diferite p și q. Aceste numere sunt utilizate pentru a calcula un produs n = p * q care este folosit ca modul (%) de criptare/decriptare
    - Se calculeaza exponentul e care este un număr întreg mai mic decat (p-1)(q-1) si care este coprim (p-1)(q-1). Aceasta va fi cheia publica
    - Cheia privată d este inversul modular al lui e mod (p-1)(q-1).

Invers modular (x * x^-1) mod N = 1

Pentru a gasi o cheie publica valida trebuie ales un număr x < (p-1)(q-1) coprim cu (p-1)(q-1)

Pentru o cheie publica (exponent) e și pentru un N fixat, pentru a afla cheia de decriptare trebuie calculat inversul modular astfel: (e * valoare) mod N = 1

13. Sistemul de criptare ElGamal

Problema logaritmului discret

-> fie G un grup ciclic de ordin q (cu |q| = n) iar g este generatorul lui G. Pentru fiecare h din G exista un unic x din Z_q astfel incat g^x = h. Problema logartimului discret cere gasirea lui x stiind G, q, g, h; notam x = log(g) h

-> poate fi rezolvata prin brute force calculand taote puterile x ale lui g pana gasim una potrivita pt care g^x = h. (O(q)).

Algoritmi generici

Metoda Baby-step/giant-step

Calculează logaritmul discret într-un grup de ordin q in timp (O (q^(½) (log q)^c).

Pohlig-Hellman

Timpul de rulare depinde de factorii primi ai lui q. Pentru ca algoritmul să nu fie eficient trebuie ca cel mai mare factor prim al lui q să fie de ordinul 2^160

Algoritmi non-generici

Potențial mai puternici decat cei generici.

Cel mai bun algoritm pentru DLP este subexponential.

Funcții hash fara coliziuni

Este posibilă obținerea unor funcții hash fără coliziuni pe baza unor prezumtii criptografice, precum PLD in grupuri de ordin prim

Fie G un grup ciclic de ordin prim și g un generator de al sau iar h un element aleator din G.
Definim H o funcție hash cu intrare de lungime fixă după cum urmează → H pentru intrarea (x1, x2) din Zq x Zq cu H(x1,x2) = g^x1 h^x2

Daca problema logaritmului discret este dificila in grupul G, atunci construcția de mai sus este o funcție hash de intrare fixa rezistenta la coliziuni.

Sistemul de criptare ElGamal

- se bazeaza pe
    - DLP (pe dificultatea problemei DDH)
    - Fie G un grup finit si m <- R G. DAca g <- R G atunci g' = mg ramane aleator in G: P (mg = g') = 1/|G| unde probabilitatea este data de alegerea aleatoare a lui g.
- dacă emițătorul și receptorul folosesc g drept cheie secretă atunci un mesaj m din G se cripteaza ca g' = mg
- receptorul decripteaza m = g' g^-1
- construcția este perfect sigură dacă g e total aleator
- in criptografia cu cheie publica g este pseudoaleator deci se pierde securitatea perfectă
- pentru ca g sa para total aleator pentru un adversar se folosește prezumția DDH

Definim sistemul de criptare ElGamal astfel:

- Se genereaza (G, q, g) și se alege x aleator din Zq si se calculeaza h = g^x
    - Cheia publica este G,q,g,h
    - Cheia privata este G,q,g,x
- Enc: data o cheie publica G,q,g,h si un mesaj m se alege y aleator din Zq și întoarce c = (c1,c2) = (g^y, m * h^y)
- Dec: dată o cheie privată G,q,g,x si un mesaj criptat c=(c1,c2) intoarce m=c2*c1^(-x)

Probleme de securitate

1. Determinism
2. Dificultatea DLP
3. Proprietatea de homomorfism
4. Utilizarea multipla a parametrilor publici

Reutilizarea valorii aleatoare introduce

1. Atacuri de repetare
2. Daca un atacator gaseste k, atunci are acces la toate mesajele
3. Atacuri de descifrari chosen plaintext (CPA)
4. Atacuri de descifrari chosen ciphertext (CCA)

14. Protocolul de schimb de chei Diffie Hellman

Un protocol de schimb de chei este un protocol prin care 2 persoane care nu partajeaza în prealabil niciun secret pot genera o cheie comuna, secreta. Acest protocol se realizeaza printr-un canal public, deci un adversar poate intercepta toate mesajele transmise pe canalul de comunicație, dar nu trebuie sa afle nimic despre cheia secreta obtinuta în urma protocolului.

Schimbul de chei Diffie-Hellman

- A si B doresc sa stabileaza o cheie secreta comuna
- A genereaza un grup ciclic G de ordin q cu |q| = n si g un generator al grupului
- A alege x <- R Zq si calculeaza h1 = g^x
- A îi trimite mesajul (G, g, q, h1) lui B
- B alege y <- R Zq si calculeaza h2 = g^y
- B îi trimite h2 lui Alice si intoarce cheia kb = h 1y
- A primește h2 și întoarce cheia ka = h 2x
- Ka = kb <=> (g^x) ^ y = (g^y) ^ x

CDH (problema de calculabilitate Diffie Hellman)

Provine de la necesitatea ca adversarul sa nu poată determina cheia comuna ka=kb chiar dacă are acces la întreaga comunicatie

Fiind date grupul ciclic G, un generator g al sau si 2 elemente h1, h2 <- R G sa se determine CDH(h1, h2) = g ^ (log(g)h1 * log(g) h2).

Rezolvarea CDH înseamnă ca adversarul determina ka=kb cunoscând h1, h2, G, g

DDH (problema de decidabilitate Diffie Hellman)

Cheia ka = kb trebuie sa fie indistinctibila fata de o valoare aleatoare

Atacul Man in the Middle

- schimbul Diffie Hellman e total nesigur pentru un adversar activ
- atacul este posibil deoarece poate impersona pe A sau pe B, de fiecare data cand A va transmite un mesaj criptat către B, O il interceptează și îl previne sa ajungă la B
- O îl decripteaza folosind kA, apoi îl recripteaza folosind kb și îl transmite către Bob
- A și B comunica fără sa fie conștienți de existența lui O

- deci A trimite h1 catre B dare este interceptat de O și transformat în h1’ (la fel și pentru h2), iar O are acces la mesaje cheile de decriptare fiind calculate în funcție de h1’ si h2’

15. Functii hash

- primesc ca argument o secvență de lungime variabila pe care o comprima într-o secvență de lungime mai mica fixată
- folosind o functie hash H pe o lista de elemente x si stocand elementul H(x) putem cauta elementul H(x) in O(1)
- în criptografie se impune minimizarea coliziunilor pentru funcțiile hash
- nu exista funcții hash fara coliziuni (functiile hash comprima, deci nu poate fi injectiva)
- se impune ca deterinarea unei coliziuni sa devina dificila

Rezistenta la coliziuni

H:{0,1}* -> {0,1}^(I(n)) unde I(n) este un polinom de gradul n, se numește rezistenta la coliziuni dacă pentru orice adversar PPT A exista o funcție neglijabilă astfel incat P(coliziune a doua hashuri) <= negl(n)

Rezistenta la a doua preimagine

Presupune ca fiind dat x este dificil de determinat x’ != x astfel incat H(x) = H(x’)

Rezistenta la prima preimagine

Presupune ca fiind dat H(x) este imposibil de determinat x

One-way function (functie calculabila intr-un singur sens)

- dat fiind y = H(x) se cere x astfel incat H(x) = y
- un atac generic necesita 2^n evaluari pentru H
- exemplu functiile hash

Securitatea functiilor hash

Rezistenta la coliziuni=> rezistența la a doua preimagine => rezistența la prima preimagine

Atacul zilei de naștere

Pentru n hashuri poți găsi o coliziune în O(2^(n/2))

Functie de compresie

Funcție hash rezistenta la coliziuni de intrare fixa.

Dacă o funcție de compresie este rezistenta la coliziuni, atunci funcția hash asociată acesteia este rezistenta la coliziuni.

Stocarea parolelor

Tipuri de atacuri:

- Atac de tip dictionar
    - adversarul poate verifica pe rand toate parolele cu probabilitate mare de apariție (exemplu de tool John the Ripper pe baza unui fișier rock_you.txt sau tool-ul medusa)
    - Pentru a bloca acest atac se blochează procesul de autentificare după un anumit număr de incercari nereusite si se obliga utilizatorul sa foloseasca o parola complexă (mărește alfabetul dicționarului)
- Atac folosind tabele hash precalculate
    - Dacă un atacator are la dispoziție hash-urile parolelor, pre calculând valorile hash ale unor parole des utilizate poate afla perechi de parole.
    - Este ineficient deoarece necesita o capacitate mare de stocare

Salting

Salt este o secvență aleatoare de n biți distinctă pentru fiecare utilizator ce se adauga la parola.
Adaugand salt-ul devine mult mai dificil ghicirea unui hash pentru un atacator.

Parole de unica folosinta

Se calculează succesiv, atacatorul nu are acces la următoarea parola dar la cele dinainte da.

MD5

- este o functie hash cu iesirea 128 biti, deci MD5 : {0,1}* -> {0,1}^128
- pasi
    - Padding
        - Mesajul este spart in blocuri de 512 biti
        - Ultimul bloc este completat pana la 448 biti cu secvența: 100..0
        - Padding-ul se realizeaza întotdeauna chiar dacă ultimul bloc are exact 448
    - Concatenarea lungimii mesajului
        - Lungimea mesajului se reprezinta pe 64 de biti (daca lungime > 2^64 se folosesc ultimii 64 de biti din reprezentarea binara)

        - Rezultatul se concateneaza la ultimul bloc care devine complet (512 biti)
        - Mesajul rezultat contine numai blocuri de 512 biti
    - Initializarea buffer-ului MD
        - Inițializarea construcției MD (Merkle Damgrad) necesita un vector de initializare pe 128 de biti
        - Acesta este sub forma unui buffer de 4 word-uri (A, B, C, D) care va prelua valori intermediare după fiecare transformare a unui bloc
    - Procesarea mesajului
        - Se realizeaza în blocuri de câte 16 word-uri
        - Compresia este de fapt o construcție Davies-Meyer unde adunarea se face modulo 2^32
        - Se definesc 4 funcții auxiliare
            - F(X,Y,Z) = daca X atunci Y altfel Z
            - G(X,Y,Z) = daca Z atunci X altfel Y
            - H(X,Y,Z) = X xor Y xor Z
            - I(X,Y,Z) = Y xor (X and (not Z))
        - Se defineste o tabela calculata folosind funcția sin
            - T[i] = 4294967296 * abs(sin(i))
        - Funcția de criptare din construcția Davies-Meyer este constituita din 4 runde, fiecare runda folosind una dintre funcțiile auxiliare F, G, H, I
    - Iesirea
        - Valoarea funcției hash este dată de valoarea finală a bufferului, mai exact are loc relatia MD(5) = ABCD

MD5 nu este sigura, iar dacă înainte era folosită pentru verificarea integrității fișierelor la descărcare (se calcula valoarea hash MD5 și se verifica dacă era identică celei inițiale), acum nu mai este sigură în practică.

SHA

- serie de functii hash care functioneaza asemanator cu MD5
- lungimea secventei de iesire poate fi pe 224 (sha0), 256 (sha256), 384 sau 512(sha512) biti
- pasi
    - Preluare
    - Divizare in blocuri (mesajele sunt divizate in blocuri de dimensiuni fixe, mai exact 512 biti pt SHA1, SHA256, SHA512 si 1024 de biti pentru SHA384)

- Padding (mesajele sunt umplute cu biti suplimentari pentru a asigura ca dimensiunea totala a blocurilor este un multiplu al dimensiunii blocurilor)
- Initializarea starii interne
- Comprimarea blocurilor (fiecare bloc este combinat cu starea interna folosind o functie de compresie rezultand o noua stare interna)
- Finalizare (starea interna finala este utilizata pentru generarea hashului final al mesajului, acest hash avand o lungime fixa si e o reprezentare unica a mesajului)

16. Semnaturi digitale

Sunt utilizate pentru a asigura autenticitatea, integritatea și neabuzabilitatea datelor digitale.
Sunt utilizate în tranzacțiile online, în autentificarea utilizatorilor și în alte aplicații care necesită verificarea originii și integrității datelor digitale.

Identificarea unei scheme de semnatura digitala nesigură

Dependența de algoritmi de criptografie vulnerabili sau de parametri slabi, utilizarea unui spațiu de chei mici, lipsa protecției împotriva atacurilor brute force sau lipsa mecanismelor adecvate de autentificare și autorizare a semnatarului

Utilizarea RSA într-o schemă de semnătură digitală:

1. RSA poate fi utilizat într-o schemă de semnătură digitală prin aplicarea algoritmului de semnare RSA.
2. Procesul de semnare digitală RSA implică generarea unei perechi de chei RSA (publică și privată), similar cu generarea de chei pentru criptarea RSA.
3. Semnarea unui mesaj cu RSA implică calcularea unei sume de control criptografice a mesajului (denumită și "hash") și aplicarea algoritmului de semnare RSA asupra acestei sume de control.
4. Pentru a verifica semnătura digitală, destinatarul utilizează cheia publică RSA a semnatarului pentru a verifica semnătura, calculând suma de control a mesajului și comparând-o cu valoarea semnăturii decodate folosind cheia publică RSA.

Crearea unei semnături RSA false:

1. Crearea unei semnături RSA false este dificilă într-un sistem corect implementat și securizat.
2. Cu toate acestea, în cazul în care un atacator obține acces la cheia privată RSA a semnatarului, acesta poate crea semnături false pentru orice mesaj.

3. De asemenea, dacă un atacator poate obține un mesaj semnat și semnătura corespunzătoare, există riscul că acesta să poată folosi aceste informații pentru a obține acces ilegal la sisteme sau pentru a falsifica autentificările.
4. Este important să se protejeze cheile private RSA și să se utilizeze practici adecvate de gestionare a cheilor pentru a preveni astfel de atacuri.

Algorithms

RSA

Steps:

1. key generation
2. encryption
3. decryption

Key Generation

1. Select two large **prime** numbers (for better security)

$$let\ p = 3\ and\ q = 11$$

2. Calculate $n = p * q \ \Rightarrow \ n = 3 * 11 = 33$
3. Calculate $\phi(n) = (p - 1)(q - 1) = 2 * 10 = 20$
4. Choose the value of e such that $1 < e < \phi(n)\ and\ gcd(\phi(n), e) = 1$

$$let\ e = 7$$

5. Calculate $d = e^{-1} mod\ \phi(n) \ \Rightarrow \ ed = 1\ mod\ \phi(n) \ \Rightarrow \ ed\ mod\ \phi(n) = 1$

$$ed\ mod\ \phi(n) \ = \ 1 \ \Rightarrow \ 7d\ mod\ 20 = 1 \Rightarrow d = 3$$

6. $pub_{key} = \{e, n\} = \{7,33\}$
7. $priv_{key} = \{d, n\} = \{3,33\}$

Encryption

$$c = m^e\ mod\ n, where\ m = number\ of\ digits\ in\ PT, where\ PT = plain\ text, m < n \ \rightarrow \ let\ m = 31$$

$$c = 31^7\ mod\ 33 = 4$$

Decryption

$$m = c^d\ mod\ n = 4^3\ mod\ 33 = 31$$

ElGamal

Steps:

1. key generation
2. encryption
3. decryption

Key Generation

1. Select large prime number $let\ p = 11$

2. Select a decryption key also called private key $let\ d = 3$
3. Select second part of encryption key $let\ e1 = 2$
4. Select third part of encryption key $let\ e2 = e1^d\ mod\ p = 8$
5. $pub_{key} = (e1, e2, p)\ and\ priv_{key} = d$

$$pub_{key} = (2,8,11)$$

Encryption

1. Select random integer R:
2. Calculate $c1 = e1^R\ mod\ p = 5$
3. Calculate $c2 = (PT * e2^R)\ mod\ p = 6$

$$where\ PT = plain\ text$$

4. $Ciphertext = (c1, c2) = (5,6)$

Decryption

1. $PT = [c2 * (c1^d)^{-1}]\ mod\ p = 1$
2.