Algoritmul criptografic RSA

Proiect

Realizat de:

Dănceanu-Chiriac Denis-Marian

Moisă Andrei

Universitatea Alexandru-Ioan Cuza Iași, Facultatea de Informatică

1. Ianuarie 2025

Cuprins

1. Introducere

* 1. Contextul criptografiei moderne
  2. Locul RSA în peisajul criptografiei moderne
  3. Relevanța pe termen lung

1. Istoric și descoperirea algoritmului RSA
   1. Origini și autori
   2. Contribuții conexe
2. Matematica din spatele RSA
   1. Problema factorizării numerelor mari
   2. Funcția totient (Euler’s totient function)
   3. Teorema resturilor chinezești
   4. Problema logaritmului discret
3. Funcționalitatea algoritmului RSA
   1. Generarea cheilor
   2. Criptare și semnătură digitală
   3. Exemplu
4. Utilizări practice ale RSA
   1. Infrastructura cu chei publice
   2. Semnături digitale și autentificare
   3. Exemple de implementări uzuale
5. Provocări actuale și criptografia post-cuantică
   1. Tehnici de criptografie post-cuantică
   2. Cum ar putea fi RSA „spart” de computerele cuantice
   3. Ultimele descoperiri despre vulnerabilități
6. Descoperiri recente și viitorul RSA
   1. Metode de optimizare a implementărilor
   2. Atacuri și contramăsuri
   3. Perspective pe termen lung
7. Partea practică
   1. Exemple de cod și demonstrație
   2. Experimente cu mărimea cheilor
   3. Analiza securității
8. Bibliografie

1. Introducere

1.1 Contextul criptografiei moderne

Criptografia a parcurs un drum lung de la metodele clasice, cum ar fi cifrul lui Caesar din Antichitate, la mecanismele computerizate actuale. În timp ce vechile sisteme de criptare se bazau adesea pe simple substituții sau transpoziții (care puteau fi sparte relativ ușor cu ajutorul statisticii frecvențelor literelor), criptografia modernă folosește concepte avansate din teoria numerelor și din algebra abstractă.

Revoluția adevărată în criptografie a început în anii ’70, când a fost introdusă ideea de criptografie asimetrică (sau cu cheie publică). Această inovație a rezolvat una dintre cele mai stringente probleme ale criptografiei simetrice: distribuirea sigură a cheii de criptare. Într-un sistem de criptare simetric, atât expeditorul, cât și destinatarul folosesc aceeași cheie pentru criptare și decriptare; prin urmare, înainte de orice comunicare, cei doi trebuiau să se asigure că își transmit cheia în condiții de maximă siguranță, ceea ce devenea complicat la scară largă.

Prin contrast, în criptografia asimetrică, fiecare utilizator are două chei: o **cheie publică**, care poate fi împărtășită oricui, și o **cheie privată**, care trebuie păstrată secretă. Oricine poate cripta date cu ajutorul cheii publice, însă doar posesorul cheii private corespunzătoare poate decripta acele date. Prin urmare, partea de distribuire a cheilor este mult simplificată, deoarece cheia publică poate fi făcută cunoscută în mod liber, fără riscuri majore de securitate.

1.2 Locul RSA în peisajul criptografiei moderne

În acest context, algoritmul RSA s-a impus rapid drept unul dintre cele mai populare și mai răspândite sisteme de criptografie asimetrică. A fost dezvoltat la MIT în 1977 și publicat în 1978, fiind pentru mult timp primul algoritm de criptare cu cheie publică care oferea nu doar criptare și decriptare, ci și posibilitatea de semnătură digitală. Într-adevăr, RSA face posibilă validarea autenticității unui mesaj, prin aceea că doar cheia privată a semnatarului poate genera semnătura, iar oricine deține cheia publică poate verifica dacă semnătura respectivă este validă.

**De ce RSA a devenit atât de răspândit?**

1. **Siguranța bazată pe factorizarea numerelor mari**: Altfel spus, este dificil (din punct de vedere computațional) să factorizezi un număr mare, de ordinul a sute de biți (astăzi chiar mii de biți), pentru a extrage primele ppp și qqq.
2. **Simplicitatea conceptuală**: Odată înțeleasă ideea de exponențiere modulară, RSA nu este dificil de implementat programatic. Prin comparație, alte scheme pot necesita structuri matematice mai sofisticate (cum ar fi curbele eliptice).
3. **Adoptare pe scară largă**: De la certificatele SSL/TLS până la sistemele de licențiere software și serviciile de e-mail securizat, RSA a devenit standard. A fost unul dintre primele algoritmi luați în considerare de către marile organizații de standardizare și a rămas până astăzi compatibil cu majoritatea bibliotecilor de criptografie.
4. **Abilitatea de a semna digital**: Pe lângă criptare, RSA este și o soluție robustă pentru semnarea mesajelor și documentelor, oferind integritate și non-repudiere.

**Importanța în infrastructura cu chei publice (PKI)**  
RSA este, totodată, un element cheie în infrastructurile de chei publice (Public Key Infrastructure). O PKI definește politici, proceduri și protocoale care permit diferitelor entități (utilizatori, servere, sisteme) să se identifice, să își verifice autenticitatea și să comunice în mod sigur. Certificatele digitale, semnate de o autoritate de certificare de încredere, conțin cheia publică a unei entități și asigură că acea cheie aparține cu adevărat entității respective. Majoritatea certificatelor TLS/SSL emise la nivel mondial se bazează pe RSA sau, în prezent, pe ECC (Curbe Eliptice) ca algoritm principal de criptare și semnătură.

**Rezistența la atacuri**  
În formele sale standard, RSA este considerat sigur atâta vreme cât dimensiunea cheii este suficient de mare pentru a face factorizarea lui n = p × q imposibilă din punct de vedere practic. Astăzi, specialiștii consideră că RSA cu cheie de 2048 de biți oferă un nivel adecvat de securitate pentru majoritatea aplicațiilor comerciale; pentru aplicații cu cerințe stricte de securitate pe termen lung, se pot folosi chei RSA de 3072 sau 4096 de biți. Cu toate acestea, există și preocupări legate de viitorul computerelor cuantice, care, teoretic, ar putea compromite RSA prin algoritmul lui Shor. Această perspectivă a generat un val de cercetări în domeniul criptografiei post-cuantice, însă, în momentul actual, RSA rămâne încă una dintre soluțiile de bază în securitatea informatică globală.

1.3 Relevanța pe termen lung

Prin faptul că este integrat în multiple standarde internaționale și suite de securitate, RSA a devenit practic un **simbol** al criptografiei moderne. Un alt motiv al popularității sale este că implementarea RSA nu necesită licențe costisitoare, deoarece algoritmul este deseori disponibil în biblioteci de criptografie open-source (de exemplu, OpenSSL). Acest lucru a permis industriei și sectorului public să adopte rapid algoritmul, indiferent de mărimea organizației sau buget.

Totuși, se discută intens dacă RSA va rezista sau nu “furtunii cuantice” și se fac cercetări ample pe algoritmi alternativi (scheme cu cheie publică bazate pe probleme din teoria grilelor – lattice-based cryptography, scheme cod-based, hash-based etc.). Până la apariția efectivă a unor computere cuantice suficient de puternice, RSA rămâne unul dintre pilonii fundamentali ai securității cibernetice și un reper în orice discuție legată de criptografia asimetrică.

2. Istoric și descoperirea algoritmului RSA

La momentul apariției sale, algoritmul RSA a oferit o soluție concretă la una dintre cele mai arzătoare probleme ale criptografiei: schimbul securizat de informații într-o manieră care să nu implice o întâlnire prealabilă sau un canal privat pentru distribuirea cheilor. Pentru a înțelege mai bine semnificația descoperirii RSA, este necesar să privim către contextul istoric, evoluțiile științifice și nevoile sociale ale epocii. În anii ’70, se putea observa o creștere rapidă a utilizării computerelor în mediul militar, guvernamental și, treptat, în sectorul privat. Deși exista interes pentru protejarea comunicațiilor, criptografia “la modă” în vremea respectivă era criptografia simetrică, bazată pe o singură cheie. Acest lucru ridica probleme complexe privind schimbul inițial de chei prin canale securizate, inaccesibile publicului larg.

Descoperirea și rafinarea tehnicilor de criptografie asimetrică (cu cheie publică) au schimbat fundamental modul în care datele puteau fi securizate. Pe lângă aspectul practic al schimbului de date, era nevoie și de validarea identității expeditorului, respectiv de modul în care destinatarul ar putea verifica rapid dacă un mesaj nu a fost modificat. Cercetările numeroase desfășurate în universități de top, inclusiv Massachusetts Institute of Technology (MIT), au urmărit să găsească soluții cât mai simple și robuste pentru această problemă.

2.1 Origini și autori

**Ronald Rivest, Adi Shamir și Leonard Adleman**  
Algoritmul RSA, botezat astfel după inițialele celor trei creatori (R, S, A), a fost dezvoltat la MIT spre sfârșitul anilor ’70. Ronald Rivest, Adi Shamir și Leonard Adleman erau pe atunci profesori și cercetători cu o puternică pregătire matematică și un interes aparte pentru noile concepte de criptografie asimetrică. Momentul cheie care a condus la descoperirea RSA a fost, în mare parte, inspirat de rezultatele lui Whitfield Diffie și Martin Hellman legate de schimbul de chei fără un canal securizat (publicate în 1976). Ideea că se poate separa cheia de criptare de cea de decriptare, folosind principii matematice avansate, i-a determinat pe Rivest, Shamir și Adleman să caute o schemă practică de implementare.

Conform relatărilor, Rivest ar fi lucrat timp de multe nopți încercând să găsească o transformare matematică perfect inversabilă, dar care să rămână dificil de inversat fără informația secretă. Punctul decisiv a venit atunci când a integrat ideea faptului că este greu să factorizezi un număr mare în doi factori primi mari. Odată identificați acești doi primi, ppp și qqq, se poate construi un algoritm care, în esență, le folosește produsul n = p x q drept „punct de blocaj” pentru un potențial atacator.

În 1977, echipa și-a finalizat cercetările, iar în 1978 a publicat oficial lucrarea în care descriau algoritmul RSA. Din acel moment, lumea criptografiei a fost puternic influențată de aceste idei, iar interesul din partea instituțiilor guvernamentale, a băncilor și a companiilor private a crescut enorm. De asemenea, au început discuțiile despre reglementarea exportului de “tehnologie criptografică” în afara Statelor Unite, deoarece securitatea comunicațiilor devenise o temă geopolitică majoră.

**Aspectul revoluționar al descoperirii**  
Prin RSA, nu doar că se securizau comunicațiile, dar se permitea și realizarea semnăturilor digitale – un element esențial pentru autentificarea expeditorului și asigurarea integrității mesajului. Această dublă funcționalitate (criptare-decriptare + semnătură digitală) a reprezentat încă un motiv pentru care RSA a căpătat rapid notorietate. Față de schema de schimb de chei propusă de Diffie-Hellman, RSA venea cu un mecanism mai direct și mai ușor de aplicat în practică pentru semnătura digitală, folosind aceeași tehnică de ridicare la putere modulară.

De-a lungul timpului, cei trei autori ai RSA au primit numeroase premii și distincții, iar în 2002, Rivest, Shamir și Adleman au fost distinși cu **Premiul Turing**, considerat echivalentul Premiului Nobel în domeniul informaticii. Lucrarea lor este considerată una dintre pietrele de temelie ale criptografiei moderne.

2.3 Contribuții conexe

**Descoperirea lui Clifford Cocks**  
În mod surprinzător, există documente care atestă faptul că o schemă similară de criptare asimetrică pe baza factorizării numerelor mari fusese deja descoperită, în 1973, de către **Clifford Cocks**, pe atunci cercetător la GCHQ (Government Communications Headquarters) în Marea Britanie. Cocks a publicat un raport intern care descria un algoritm apropiat de ce avea să devină RSA, însă lucrarea a rămas clasificată. Din cauza secretelor guvernamentale, informațiile nu au fost făcute publice, astfel că meritele descoperirii publice au revenit, în mod firesc, echipei RSA de la MIT.

Abia la mijlocul anilor ’90, GCHQ a desecretizat parțial documentele, dezvăluind că au existat cercetări similare încă de la începutul deceniului ’70. În pofida acestui fapt, recunoașterea internațională a algoritmului RSA aparține fondatorilor săi oficiali, iar RSA a fost primul algoritm de criptare asimetrică distribuit pe scară largă în mediul civil și academic.

**Influența protocolului Diffie-Hellman**  
Deși RSA și Diffie-Hellman sunt algoritmi cu filozofii ușor diferite (RSA se bazează pe factorizarea numerelor mari, iar Diffie-Hellman pe problema logaritmului discret în grupuri finite), apariția și publicarea protocolului de schimb de chei propus de Whitfield Diffie și Martin Hellman au servit ca sursă de inspirație. Ei au fost primii care au demonstrat cum criptografia asimetrică poate rezolva problema transmisiei cheilor prin canale nesigure, fapt ce i-a motivat și pe Rivest, Shamir și Adleman să caute o schemă completă (criptare + semnătură). Altfel spus, existența unei idei de criptografie asimetrică a servit drept “dovadă” că tehnologia ar putea funcționa, impulsionând dezvoltarea unui mecanism la fel de puternic, bazat însă pe alte probleme dificile (factorizarea).

**Impactul asupra criptografiei ulterioare**  
Odată cu succesul RSA, preocuparea pentru analiza problemelor matematice dificile (hard problems) a crescut. Au urmat cercetări intense în jurul altor tipuri de probleme, precum:

* **Logaritmul discret** (care stă la baza protocolului ElGamal, a standardului DSA și a criptografiei cu curbe eliptice).
* **Problema SIS (Short Integer Solution) și LWE (Learning With Errors)**, care sunt fundamentale pentru schemele post-cuantice bazate pe structuri de tip lattice.

RSA a demonstrat că un algoritm de criptare poate fi sigur și practicat pe scară largă, deschizând porțile către o nouă eră a comunicațiilor securizate. Acest fapt a încurajat guvernele și marile companii să investească masiv în soluții de securitate criptografică și să promoveze standarde de protecție a datelor. Astfel, criptografia nu mai era un domeniu de nișă, limitat doar la spionaj militar sau diplomatic, ci devenea o componentă esențială a vieții cotidiene digitale.

3. Matematica din spatele RSA

Algoritmul RSA își datorează securitatea și funcționalitatea unei serii de concepte matematice fundamentale din teoria numerelor. În centrul acestui sistem criptografic se află **factorizarea numerelor mari** și proprietățile funcției totient a lui Euler, însă o contribuție semnificativă la eficiența implementării și la asigurarea securității o au și **Teorema Resturilor Chinezești** și înțelegerea **problemei logaritmului discret**. Fiecare dintre aceste elemente furnizează o „piesă” esențială din „puzzle-ul” RSA, conferind rezistență la atacuri brute-force și imposibilitatea practică (în prezent) de a deduce cheia privată din cheia publică.

3.1 Problema factorizării numerelor mari

**Definiție și importanță**  
Problema factorizării numerelor mari constă în a găsi factorii primi care compun un număr întreg n. Cu alte cuvinte, dacă nnn este un produs de forma n = p x q, unde p și q sunt numere prime mari, atunci întrebarea este: cât de greu îi este unui atacator să recupereze acești factori p și q? În mod obișnuit, cu cât p și q sunt mai mari, cu atât factorii lor vor fi mai dificil (și mai costisitor ca timp de calcul) de găsit.

RSA se bazează direct pe această dificultate: cheia publică divulgă numărul n, dar nu și factorii p și q. Dacă un adversar nu poate factorizan, el nu poate afla funcția totient) φ(n) și, implicit, nu poate deduce exponentul de decriptare d. Aceasta face ca RSA să fie considerat sigur pentru chei cu lungime de 2048 de biți (sau mai mari), în condițiile tehnologiei convenționale actuale.

**Metode de factorizare și starea curentă**

1. **Fermat’s Factorization Method**: Eficientă doar atunci când factorii ppp și qqq sunt relativ apropiați, ceea ce, în implementarea RSA, se evită în mod intenționat.
2. **Pollard’s Rho**: O metodă probabilistică de factorizare care poate găsi factori relativ mici, dar devine ineficientă la dimensiuni de ordinul miilor de biți.
3. **Metoda GNFS (General Number Field Sieve)**: Considerată în prezent cea mai rapidă metodă de factorizare a numerelor foarte mari, însă și aceasta are o complexitate care crește substanțial cu mărimea cheii.

Cât despre „spargerea” RSA prin factorizarea directă a lui nnn, încercările curente folosesc uriașe resurse de procesare (uneori rețele cu mii de computere) pentru a factoriza numere RSA cu dimensiuni de ~768 biți, reușind acest lucru în luni sau chiar ani. De aceea, standardul comercial actual a migrat spre 2048 de biți sau mai mult, considerând că atacurile de factorizare directă sunt în continuare deosebit de dificile și neprofitabile pentru atacatori.

**Relevanța pentru RSA**

* **Confidențialitate**: Dacă un adversar nu poate factoriza n, nu poate afla d – exponentul privat.
* **Semnătură digitală**: Pentru falsificarea unei semnături, ar fi necesar același exponent privat d, deci iarăși e nevoie de factorii lui n.
* **Robustețe împotriva atacurilor brute-force**: Atât timp cât problema factorizării rămâne nerezolvată și considerată a fi dincolo de resursele de calcul disponibile, RSA rămâne viabil.

3.2 Funcția totient (Euler’s totient function)

**Definiție și proprietăți**  
Funcția totient a lui Euler, notată phi(n), reprezintă numărul de întregi pozitive mai mici decât n care sunt coprime cu n (adică nu au niciun divizor comun mai mare decât 1 cu n). Iată câteva proprietăți importante:

* Dacă p este număr prim, atunci phi(p) = p - 1.
* Dacă n și m sunt coprime, atunci phi(n \* m) = phi(n) \* phi(m).
* În special, dacă n = p \* q, cu p și q prime, atunci phi(n) = (p - 1) \* (q - 1).

Aceste proprietăți sunt simple, dar au implicații majore în criptografie. Motivul pentru care phi(n) joacă un rol crucial în RSA este că exponentul privat d se definește prin relația:  
e \* d ≡ 1 (mod phi(n)).

Această condiție asigură că ridicarea la putere cu e și apoi cu d (sau invers) „anulează” efectul una alteia, redând mesajul inițial.

**Interpretarea în criptarea RSA**

* **Generarea cheilor**: După ce am ales primele p și q, calculăm n = p \* q și phi(n) = (p - 1) \* (q - 1). Ulterior, selectăm un exponent public e, care trebuie să fie coprime cu phi(n). Odată stabilit e, determinăm d, astfel încât d să fie inversul modular al lui e, calculat modulo phi(n).
* **Fundament matematic**: Este necesară cunoașterea lui phi(n) pentru a construi relația de inversare în operațiile de ridicare la putere modulară. Fără phi(n), nu putem stabili unicitatea inversului modular pentru e.

Întreaga „magie” din RSA se bazează pe faptul că un atacator nu poate deduce phi(n) (și implicit d) fără să cunoască factorii lui n.

3.3 Teorema resturilor chinezești

**Context și formulare**  
Teorema Resturilor Chinezești afirmă că, pentru un set de congruențe:

x ≡ a1 (mod m1)  
x ≡ a2 (mod m2)

unde m1 și m2 sunt coprime, există o soluție unică pentru x în raport cu M = m1 \* m2, iar această soluție poate fi calculată eficient.

**Relevanța pentru RSA**  
Direct, CRT nu face parte din algoritmul de bază RSA, însă este esențială pentru **optimizarea** implementării. Când realizăm decriptarea, c^d mod n, putem calcula separat:

m\_p = c^d mod p și  
m\_q = c^d mod q,

unde p și q sunt factorii lui n. Aceste calcule sunt mult mai rapide, pentru că p și q sunt mult mai mici decât n. Apoi, combinăm rezultatele m\_p și m\_q cu ajutorul CRT pentru a obține mesajul m în mod unic.

**Avantajele practice**

* **Reducerea timpului de calcul**: Pentru chei RSA mari, decriptarea și semnarea pot fi accelerate semnificativ (de multe ori de câteva ori).
* **Implementări robuste**: CRT permite optimizări care exploatează din plin resursele hardware (de exemplu, în smartcard-uri sau în HSM – Hardware Security Modules).
* **Utilizare pe scară largă**: Majoritatea bibliotecilor de criptografie (OpenSSL, Crypto++ etc.) folosesc CRT pentru a accelera calculele de ridicare la putere modulară.

3.4 Problema logaritmului discret

**Definiție și analogie**  
Problema logaritmului discret (Discret Logarithm Problem – DLP) se referă la dificultatea de a rezolva ecuația:

g^x ≡ h (mod p),

atunci când sunt cunoscute valorile g, h și p, dar se dorește aflarea lui x. Acest tip de problemă apare în mod special în protocolul Diffie-Hellman și în scheme criptografice precum ElGamal ori DSA.

**Legătura cu RSA**  
Deși RSA nu se bazează pe problema logaritmului discret, ambele se referă la imposibilitatea (în practică) de a inversa rapid o operație de ridicare la putere modulară:

* RSA se bazează pe **factorizarea** numerelor mari.
* Alte sisteme (Diffie-Hellman, ElGamal, DSA) se bazează pe **logaritmul discret**.

Ambele familii de probleme sunt considerate „hard problems” și stau la baza securității criptografiei moderne cu cheie publică. Faptul că există două tipuri diferite de probleme greu de rezolvat oferă diversitate și siguranță suplimentară împotriva eventualelor progrese matematice sau tehnologice care ar putea rezolva eficient una dintre ele.

**Diferențe conceptuale**

* **Factorizarea vs. Logaritm discret**: Factorizarea se referă la descompunerea unui număr în factorii săi, în timp ce logaritmul discret se referă la identificarea exponentului dintr-o congruență modulară.
* **Implementări criptografice**: RSA utilizează ridicarea la putere modulară cu exponenți mari și funcția totient, în timp ce sistemele cu logaritm discret se bazează pe generatoare și grupuri finite.
* **Rezistența la atacuri cuantice**: Atât factorizarea, cât și logaritmul discret pot fi rezolvate, teoretic, de un computer cuantic (prin algoritmul lui Shor), ceea ce ar putea amenința securitatea actualelor sisteme bazate pe aceste probleme.

4. Funcționalitatea algoritmului RSA

Algoritmul RSA a devenit un standard în criptografia modernă deoarece oferă atât mecanisme de criptare-decriptare, cât și suport pentru semnătura digitală. Acesta se bazează pe ideea de a separa cheia de criptare (publică) de cheia de decriptare (privată), folosind proprietăți ale funcției totient și dificultatea factorizării numerelor mari. În cele ce urmează, vom descrie în detaliu cum se generează cheile RSA, cum se realizează criptarea și semnarea digitală, precum și un exemplu practic (pe valori mici) care ilustrează pașii cheie ai algoritmului.

4.1 Generarea cheilor

Pașii de generare a cheilor RSA sunt concepuți astfel încât să asigure atât securitatea, cât și ușurința în utilizare. Ideea principală este următoarea: se aleg două numere prime mari, p și q, apoi se construiește n = p \* q și se definește phi(n) = (p - 1) \* (q - 1). Din aceste valori se aleg exponenții e (public) și d (privat), cu anumite proprietăți matematice ce facilitează criptarea și decriptarea. Iată procesul detaliat:

1. **Alegerea numerelor prime p și q**
   * Se aleg p și q suficient de mari (de obicei, între 1024 și 4096 de biți pentru fiecare).
   * În mod obișnuit, se folosesc teste probabilistice de primalitate, precum testul Miller-Rabin, care oferă o garanție foarte mare că numerele generate sunt prime.
2. **Calculul lui n**
   * Se calculează n = p \* q.
   * Aceasta este o operație directă, însă factorii săi, p și q, trebuie păstrați secreți.
3. **Calculul funcției totient**
   * Se calculează phi(n) = (p - 1) \* (q - 1).
   * phi(n) va fi necesar pentru stabilirea exponenților.
4. **Alegerea exponentului public e**
   * e trebuie să fie coprime cu phi(n). Cu alte cuvinte, gcd(e, phi(n)) = 1.
   * În practică, se alege adesea e = 65537 (0x10001), fiind un exponent suficient de mare pentru securitate, dar totodată optim pentru calcule rapide.
5. **Calculul exponentului privat d**
   * d este inversul modular al lui e modulo phi(n), adică trebuie să satisfacă:  
     e \* d ≡ 1 (mod phi(n)).
   * Există algoritmi rapizi (precum Extended Euclidean Algorithm) care pot găsi d în timp eficient.
6. **Cheia publică și cheia privată**
   * **Cheia publică** este formată din perechea (n, e) și poate fi distribuită oricui.
   * **Cheia privată** este formată din (n, d), în sensul că doar deținătorul acestei perechi are cunoștință de d (și, bineînțeles, de valorile p și q inițiale).

Rezultatul final este că oricine poate folosi cheia publică (n, e) pentru a cripta mesaje sau pentru a verifica semnături, în timp ce numai deținătorul cheii private (n, d) poate decripta sau semna mesaje.

4.2 Criptare și semnătură digitală

O caracteristică unică și foarte utilă a RSA este faptul că, folosind aceleași seturi de chei, se pot realiza atât criptarea datelor, cât și semnătura digitală. Diferența constă în ordinea în care este folosită cheia (publică sau privată) pentru ridicarea la putere modulară.

1. **Criptare**
   * Să spunem că Alice vrea să trimită un mesaj M către Bob, în mod confidențial.
   * Bob îi furnizează lui Alice cheia sa publică: (n, e).
   * Alice transformă mesajul M (de obicei reprezentat sub formă numerică sau bloc de biți) în C folosind:  
     C = M^e mod n.
   * Textul cifrat C poate fi transmis în siguranță către Bob, deoarece numai cheia privată (n, d) poate inversa transformarea.
2. **Decriptare**
   * Pentru a recupera mesajul M din C, Bob va folosi cheia sa privată (n, d) și va calcula:  
     M = C^d mod n.
   * Operația de ridicare la putere modulară cu d „anulează” operația efectuată cu e, întrucât e \* d ≡ 1 (mod phi(n)).
3. **Semnătura digitală**
   * Pentru a semna un mesaj, expeditorul (de exemplu, Alice) folosește **cheia sa privată** (n, d).
   * Procesul tipic presupune întâi generarea unui hash al mesajului, notat H(M). Semnarea directă a întregului mesaj ar fi ineficientă în multe cazuri, astfel că se folosește un rezumat (hash).
   * Semnătura S se obține prin:  
     S = (H(M))^d mod n.
   * Odată ce semnătura S este creată, Alice transmite atât M, cât și S către Bob.
4. **Verificarea semnăturii**
   * Bob folosește cheia publică a lui Alice, (n, e), pentru a verifica autenticitatea semnăturii.
   * Calculează: H’ = S^e mod n.
   * Dacă H’ coincide cu H(M) (calculat separat de Bob direct din mesaj), atunci Bob știe că semnătura este validă și că Alice a semnat mesajul.
   * În plus, această verificare a semnăturii asigură integritatea mesajului, deoarece orice modificare ar altera hash-ul și ar duce la un eșec în validare.

Prin acest mecanism dual, RSA poate asigura confidențialitatea (prin criptare) și autenticitatea/integritatea (prin semnătura digitală) într-o singură infrastructură de chei.

4.3 Exemplu

Pentru a ilustra pașii algoritmici, vom folosi un exemplu cu valori mici, departe de a fi sigur în practică, dar suficient pentru a înțelege mecanica RSA. Să spunem că alegem:

* p = 7
* q = 13

1. **Calculul lui n**  
   n = p \* q = 7 \* 13 = 91
2. **Calculul lui phi(n)**  
   phi(n) = (7 - 1) \* (13 - 1) = 6 \* 12 = 72
3. **Alegerea lui e**  
   Alegem e = 5 (trebuie să fie coprime cu 72). Aici, gcd(5, 72) = 1, deci este valid.
4. **Calculul lui d**  
   Avem nevoie de d astfel încât 5 \* d ≡ 1 (mod 72).  
   Prin calcule (sau folosind algoritmi de invers modular), se găsește d = 29, fiindcă 5 \* 29 = 145, iar 145 mod 72 = 1.
5. **Cheia publică și Cheia privată**
   * Cheia publică: (n, e) = (91, 5).
   * Cheia privată: (n, d) = (91, 29).
6. **Criptare**  
   Presupunem că vrem să criptăm mesajul M = 9.  
   C = 9^5 mod 91.  
   9^5 = 59049, iar 59049 mod 91 poate fi calculat pas cu pas:  
   59049 mod 91 = 63 (folosind calcule repetitive sau un calculator).  
   Deci C = 63.
7. **Decriptare**  
   M = 63^29 mod 91.  
   Deși pare complicat, efectuat modular, rezultatul final ar trebui să revină la 9.
8. **Semnătură (exemplu simplist)**
   * Dacă vrem să semnăm M = 9 folosind cheia privată (91, 29):  
     S = 9^29 mod 91.
   * Pentru a verifica, oricine are cheia publică (91, 5) poate calcula S^5 mod 91 și ar trebui să obțină 9 (sau un hash asociat, dacă am folosi hashing).

Deși exemplele reale folosesc numere mult mai mari și includ diverse scheme de padding (pentru a evita atacurile bazate pe mesaje alese), acest prototip arată clar logica din spatele RSA.

5. Utilizări practice ale RSA

5.1 Infrastructura cu Chei Publice (PKI)

RSA este piatra de temelie a infrastructurii cu chei publice (PKI), un sistem fundamental pentru securizarea comunicațiilor digitale. PKI este utilizat pentru gestionarea, distribuirea și validarea certificatelor digitale, care conțin cheia publică RSA, fiind esențiale pentru criptare, semnături digitale și autentificare. Acest sistem asigură trei piloni ai securității cibernetice: confidențialitate, integritate și autenticitate.

Exemple concrete de utilizări ale PKI și RSA includ:

* **Protocolul HTTPS/TLS**: RSA este utilizat pentru negocierea unui canal securizat de comunicare între browser și server. De exemplu, atunci când accesezi un site web securizat, certificatul digital al serverului validează identitatea acestuia, iar cheia RSA ajută la schimbul de chei simetrice utilizate ulterior pentru criptarea sesiunii. Acest proces protejează informații sensibile, cum ar fi datele de autentificare sau detaliile financiare.
* **E-mail securizat (S/MIME)**: Tehnologia RSA este aplicată pentru criptarea e-mailurilor și adăugarea de semnături digitale. Aceasta garantează că doar destinatarul intenționat poate citi mesajul și că expeditorul este autentic. Organizații mari, inclusiv instituții guvernamentale, utilizează S/MIME pentru a asigura integritatea corespondenței lor.
* **Semnături digitale**: Cu RSA, documentele electronice pot fi semnate digital, conferindu-le valoare legală echivalentă cu semnăturile fizice. De exemplu, platformele de contractare digitală folosesc RSA pentru a valida identitatea semnatarilor și integritatea documentelor.
* **Autentificarea utilizatorilor**: Sistemele de autentificare multi-factor utilizează RSA pentru a crește nivelul de securitate. De exemplu, token-urile hardware precum YubiKey generează chei RSA unice care permit autentificarea utilizatorilor în medii sensibile, cum ar fi bănci sau rețele corporative.

În plus, RSA este critic pentru domenii emergente, cum ar fi:

* **Internetul lucrurilor (IoT)**: Dispozitivele IoT folosesc RSA pentru a asigura schimbul securizat de date între senzori, gateway-uri și cloud, prevenind atacurile cibernetice.
* **Comunicarea militară**: RSA este integrat în sistemele criptografice militare pentru a proteja comunicațiile clasificate, asigurând că doar destinatarii autorizați pot descifra mesajele transmise.

Aceste exemple demonstrează versatilitatea PKI și a criptografiei RSA, fundamentând securitatea digitală într-o varietate de domenii critice.

5.2 Semnături digitale și autentificare

Semnăturile digitale și autentificarea sunt două dintre cele mai importante aplicații practice ale RSA. Aceste tehnologii sunt utilizate pe scară largă pentru a verifica identitatea unui utilizator sau integritatea unui mesaj, având aplicații esențiale în diverse domenii.

* **Semnături digitale**: Cu ajutorul cheii private RSA, expeditorul poate crea o semnătură unică pentru un mesaj. Aceasta este dependentă de conținutul mesajului și garantează autenticitatea acestuia. Receptorul, utilizând cheia publică corespunzătoare, poate verifica dacă semnătura este validă și dacă mesajul nu a fost modificat. Exemple practice includ:
  + **Sisteme de sănătate**: În gestionarea fișelor medicale electronice, semnăturile digitale asigură că documentele nu au fost alterate și că provin de la un medic autorizat. De exemplu, rețetele digitale semnate electronic sunt validate de farmaciști înainte de eliberarea medicamentelor.
  + **Procese juridice**: Semnăturile digitale sunt utilizate pentru autentificarea documentelor legale, cum ar fi contractele sau hotărârile judecătorești. Acestea permit părților implicate să aibă certitudinea că documentele sunt autentice și că nu au suferit modificări neautorizate.
  + **Platforme de e-guvernare**: În multe țări, cetățenii pot semna documente oficiale, precum declarații fiscale, folosind semnături digitale, reducând astfel necesitatea interacțiunilor fizice.
* **Autentificare**: RSA joacă un rol central în asigurarea accesului sigur la sisteme critice și servicii online. Exemple relevante includ:
  + **Sisteme medicale**: În spitale și clinici, autentificarea bazată pe RSA este utilizată pentru a proteja accesul la informațiile sensibile ale pacienților. Medicii și personalul autorizat folosesc chei private pentru a accesa dosarele medicale și a garanta confidențialitatea acestora.
  + **Instituții juridice**: Avocații și judecătorii utilizează sisteme de autentificare bazate pe RSA pentru a accesa dosarele digitale, protejând astfel informațiile confidențiale și asigurând că numai persoanele autorizate pot vizualiza sau edita aceste documente.

Aceste exemple evidențiază modul în care semnăturile digitale și autentificarea bazate pe RSA contribuie la îmbunătățirea securității și eficienței în domenii critice, precum sănătatea și justiția.

* **Semnături digitale**: Cu ajutorul cheii private RSA, expeditorul poate crea o semnătură unică pentru un mesaj, semnătura fiind dependentă de conținutul mesajului. Receptorul, utilizând cheia publică corespunzătoare, poate verifica dacă semnătura este autentică și dacă mesajul nu a fost modificat. De exemplu, semnăturile digitale sunt folosite în:
  + Contracte digitale: Documentele semnate electronic au aceeași valoare legală ca cele semnate fizic, dacă sunt utilizate semnături digitale bazate pe RSA.
  + Sistemele de vot online: Semnăturile digitale asigură că voturile sunt autentice și că nu au fost alterate.
* **Autentificare**: RSA este folosit în autentificarea utilizatorilor pentru acces la sisteme critice sau servicii online. De exemplu:
  + **Banking online**: Multe bănci folosesc chei RSA pentru a autentifica clienții în platformele lor online, asigurând astfel securitatea tranzacțiilor.
  + **Aplicații de e-commerce**: Platformele de cumpărături online utilizează RSA pentru autentificarea și protecția datelor utilizatorilor în timpul tranzacțiilor.

Aceste aplicații subliniază importanța RSA în protejarea comunicărilor și datelor digitale, asigurând un mediu sigur pentru utilizatorii din întreaga lume.

5.3 Exemple de implementări uzuale

Există numeroase implementări ale RSA care sunt utilizate pe scară largă, fiecare având particularități și scopuri specifice. De asemenea, companii mari precum Google și Microsoft integrează RSA în infrastructurile lor pentru a garanta securitatea utilizatorilor și a datelor sensibile:

* **OpenSSL**: Această librărie open-source este una dintre cele mai cunoscute implementări ale RSA. Este folosită la scară globală pentru securizarea serverelor web prin protocolul HTTPS. Google și alte companii tehnologice o utilizează în infrastructurile lor pentru a asigura comunicarea securizată între servere și clienți. OpenSSL permite generarea de chei, semnături digitale și criptare avansată.
* **Crypto++**: O librărie completă care include o implementare optimizată a RSA, utilizată atât în cercetare, cât și în dezvoltare. Datorită flexibilității sale, Crypto++ este adesea folosită de companii pentru dezvoltarea de soluții personalizate de securitate cibernetică. Microsoft, de exemplu, o consideră o sursă de inspirație pentru unele tehnologii interne.
* **Librării standard**: Limbaje de programare populare precum Python, Java și C# oferă suport pentru RSA prin module dedicate, fiind utilizate de companii mari pentru construirea aplicațiilor enterprise și de consum:
  + **Python**: Modulul cryptography este folosit de multe platforme de e-commerce pentru a securiza datele utilizatorilor în timpul tranzacțiilor. Google integrează module similare pentru securizarea produselor sale, cum ar fi Google Cloud Platform.
  + **Java**: Biblioteca javax.crypto este utilizată pentru dezvoltarea aplicațiilor enterprise care necesită criptare asimetrică. Microsoft Azure, o platformă de servicii cloud, folosește librării similare pentru a securiza fluxurile de date.
  + **C#**: Spațiul de nume System.Security.Cryptography include funcționalități RSA folosite în mod frecvent în aplicațiile enterprise dezvoltate pentru Windows. Aplicațiile Microsoft Office integrează astfel de funcții pentru a proteja fișierele și documentele utilizatorilor.

Exemple din infrastructura acestor companii demonstrează că RSA este un pilon esențial pentru securitatea digitală, fiind utilizat în protocoale de criptare, autentificare și protecție a datelor sensibile, inclusiv în medii cloud și servicii online.

6. Provocări Actuale și Criptografia Post-Cuantică

6.1 Tehnici de criptografie post-cuantică

Criptografia post-cuantică este o ramură emergentă a securității informatice care își propune să protejeze datele împotriva amenințărilor pe care le-ar putea aduce computerele cuantice. Computerele cuantice, utilizând qubits, pot efectua calcule în mod fundamental diferit față de computerele clasice, permițând viteze de procesare mult mai rapide pentru anumite operațiuni. Criptografia tradițională, precum RSA și ECC (Elliptic Curve Cryptography), se bazează pe dificultatea unor sarcini matematice specifice, cum ar fi factorizarea numerelor mari sau rezolvarea ecuațiilor elliptice. Cu toate acestea, computerele cuantice dispun de algoritmi, precum algoritmul lui Shor, care sunt capabili să rezolve aceste probleme mult mai eficient. Astfel, devine imperativă dezvoltarea de algoritmi de criptografie care să reziste amenințărilor aduse de aceste tehnologii emergente.

Printre principalele tehnici de criptografie post-cuantică utilizate astăzi se numără:

1. **Criptografia bazată pe laturi (Lattice-based cryptography)**: Lattice-based cryptography se bazează pe probleme matematice ce implică structuri geometrice complexe, numite laturi. Aceste structuri sunt extrem de robuste împotriva atacurilor cuantice datorită dificultății crescute exponențial pentru a găsi soluții la problemele laturilor pe măsură ce acestea devin mai mari. Este considerată una dintre cele mai promițătoare tehnici de criptografie pentru viitorul apropiat, având multiple avantaje, inclusiv scalabilitate și rezistență la metodele de atac cuantice.
2. **Criptografia bazată pe funcții hash (Hash-based cryptography)**: Această metodă folosește funcții cryptografice pentru a proteja integritatea datelor și este folosită adesea pentru generarea cheilor și protejarea mesajelor. Funcțiile hash se bazează pe principii matematice complexe, care fac dificilă modificarea rapidă a valorilor lor, chiar și pentru computerele cuantice. Tehnologia hash este adesea combinată cu metode de criptare tradiționale pentru a proteja datele împotriva atacurilor viitoare.
3. **Criptografia bazată pe coduri (Code-based cryptography)**: Această tehnică folosește proprietățile codurilor corectoare de erori pentru a construi scheme de criptare sigure. Dificultatea de a sparge aceste coduri este înaltă datorită complexității matematice implicate și a necesității de a efectua operații corecte care sunt greu de realizat chiar și pentru computere clasice.
4. **Criptografia multivariată (Multivariate polynomial cryptography)**: Această tehnică se bazează pe rezolvarea ecuațiilor polinomiale multivariabile. Dificultatea de a rezolva aceste ecuații crește exponențial pe măsură ce adâncimea și dimensiunea ecuațiilor cresc. Datorită complexității lor, aceste ecuații sunt extrem de greu de atacat cu ajutorul computelor cuantice.

Toate aceste metode sunt destinate să facă față amenințărilor pe termen lung și să protejeze datele critice împotriva atacurilor emergente.

*6.2 Cum ar putea fi RSA „spart” de computerele cuantice*

RSA este unul dintre cele mai utilizate algoritmi de criptare publică în întreaga lume, fiind folosit în mod extensiv pentru a securiza comunicațiile și stocarea datelor. Totuși, RSA se bazează pe dificultatea factorizării numerelor mari, o problemă care devine extrem de simplă pentru computerele cuantice datorită algoritmului lui Shor. Algoritmul lui Shor folosește qubits pentru a efectua rapid operații de factorizare, permițând identificarea factorilor unui număr cu o eficiență deosebită. Acest lucru înseamnă că RSA ar putea fi spart într-un timp relativ scurt de computerele cuantice.

Vulnerabilitățile RSA sunt atât în fața atacurilor clasice cât și a celor cuantice:

* **Algoritmi clasici de factorizare**: Există algoritmi clasici precum GNFS (General Number Field Sieve) care devin din ce în ce mai puternici și pot realiza factorizarea numerelor mari mult mai rapid decât în trecut. Acest lucru face ca RSA să fie vulnerabil chiar și în absența calculatoarelor cuantice.
* **Atacurile side-channel**: Aceste atacuri exploatează variabilele externe cum ar fi timpul de execuție sau consumul de energie al unui dispozitiv pentru a extrage informații. RSA este adesea vulnerabil la astfel de metode de atac, permițând extragerea cheilor secrete din timpul operațiunilor criptografice.

De asemenea, RSA este susceptibil la atacuri de combinare, care implică utilizarea simultană a metodelor de factorizare și a informațiilor extrase din alte surse, ceea ce face spargerea acestuia mult mai ușoară.

Pe măsură ce computerele cuantice avansează, riscul spargerii RSA devine din ce în ce mai real. Acest lucru subliniază importanța dezvoltării unor scheme post-cuanti care să ofere protecție pe termen lung.

*6.3 Ultimele descoperiri despre vulnerabilități*

De-a lungul anilor, cercetătorii au descoperit noi metode pentru a exploata vulnerabilitățile RSA, punând în evidență nevoia urgentă de a adopta metode post-cuanti:

* **Algoritmi side-channel**: Descoperirile recente arată că atacurile care exploatează diferențele de consum de energie sau timpul de execuție devin tot mai precise și eficiente. Aceste metode reprezintă o amenințare serioasă, permițând extragerea cheilor secrete din operațiunile criptografice.
* **Atacuri combinate**: Acestea combină metodele de factorizare cu informațiile extrase din alte surse pentru a construi atacuri eficiente și rapide asupra RSA. Exploatarea simultană a diverselor vulnerabilități face ca RSA să fie extrem de periculos și ușor de spart.

Toate aceste descoperiri accentuează necesitatea de a trece la algoritmi post-cuanti, care să protejeze datele împotriva amenințărilor emergente. Fără o astfel de adaptare, multe dintre sistemele de comunicații și stocare actuale sunt extrem de vulnerabile.

**7. Descoperiri Recente și Viitorul RSA**

*7.1 Metode de optimizare a implementărilor RSA*

De-a lungul timpului, au fost dezvoltate și aplicate numeroase metode de optimizare pentru a îmbunătăți eficiența implementărilor RSA, având în vedere că algoritmul este unul dintre cele mai utilizate în securitatea datelor. Această optimizare se concentrează pe accelerarea operațiunilor de criptare și decriptare, reducerea consumului de resurse și creșterea vitezei fără a compromite securitatea.

**Teorema Resturilor Chinezești (CRT)** este una dintre cele mai importante metode de optimizare folosită în implementarea RSA. Aceasta permite realizarea de operații mai eficiente de decriptare combinând rezultatele parțiale ale factorizării. Practic, cheia privată este împărțită în două numere mai mici, d1d\_1d1​ și d2d\_2d2​, care sunt mai ușor de gestionat și prelucrat decât cheia privată completă. Decriptarea poate fi astfel realizată într-un timp semnificativ mai scurt, deoarece computațiile necesare sunt împărțite între două entități, ceea ce scurtează timpul necesar pentru găsirea inverselor modulului nnn.

O altă îmbunătățire a fost realizată prin utilizarea hardware-ului specializat pentru criptografie. Procesarea RSA devine mult mai rapidă atunci când este efectuată pe cipuri dedicate, cum ar fi Acceleratoarele de Criptografie. Aceste cipuri sunt optimizate pentru operații matematice complexe, precum factorizarea și modularea, care sunt esențiale pentru RSA. Implementările hardware sunt folosite frecvent în servere și dispozitive care au nevoie de o performanță ridicată în ceea ce privește criptarea, permițând utilizarea mai eficientă a resurselor.

Librăriile criptografice, precum OpenSSL și Crypto++, joacă, de asemenea, un rol important în optimizarea implementărilor RSA. Acestea oferă coduri bine testate și optimizate pentru operațiunile criptografice, asigurând că procesarea este rapidă și corectă. Librăriile sunt actualizate constant pentru a include cele mai noi metode de protecție împotriva atacurilor și pentru a maximiza eficiența algoritmilor RSA.

Prin toate aceste metode, implementările RSA au evoluat pentru a deveni mai eficiente și mai rapide, asigurându-se că rămân în pas cu cerințele moderne de securitate.

*7.2 Atacuri și contramăsuri împotriva RSA*

Atacurile asupra RSA au devenit din ce în ce mai sofisticate pe măsură ce tehnologia a avansat, dar și măsurile de protecție s-au adaptat în consecință. Unul dintre cele mai periculoase tipuri de atacuri este **atacul side-channel**. Acestea exploatează variațiile minore ale semnalelor electrice, ale timpului de execuție sau ale consumului de energie al procesorului pentru a extrage informații cruciale din operațiunile RSA. Astfel de atacuri sunt dificile de detectat, dar pot fi limitate prin implementări criptografice care minimizează variațiile și monitorizarea atentă a consumului de energie.

Pentru a preveni aceste tipuri de atacuri, au fost dezvoltate și implementate **schemele de padding**, cum ar fi OAEP (Optimal Asymmetric Encryption Padding). OAEP adaugă un strat suplimentar de protecție în procesul de criptare, care garantează că mesajele sunt criptate de o manieră astfel încât să fie rezistente la manipulări din partea adversarilor care încearcă să extragă conținutul mesajelor. De asemenea, schemele de padding au fost concepute să reziste atacurilor de tip text-criptat ales, un tip de atac în care atacatorul poate alege texte particulare pentru a observa răspunsurile criptografice și pentru a încerca să extragă cheile secrete.

Prin aceste măsuri, sistemele RSA pot oferi o protecție mai bună împotriva atacurilor sofisticate, dar este important ca implementările să fie realizate corespunzător și să fie actualizate periodic pentru a se adapta la noi tipuri de atacuri.

*7.3 Perspective pe termen lung: RSA și viitorul criptografic*

Pe măsură ce tehnologiile avansează și computerele cuantice devin din ce în ce mai puternice, viitorul algoritmilor de criptografie se îndreaptă inevitabil către implementarea unor scheme **post-cuanti**. RSA, în forma sa actuală, este extrem de vulnerabil la computerele cuantice, mai ales datorită algoritmului lui Shor, care poate factoriza rapid numerele mari. Aceasta înseamnă că RSA va fi înlocuit treptat, pe măsură ce computerele cuantice devin disponibile, iar standardele criptografice se vor adapta pentru a asigura securitatea pe termen lung.

**Schemele hibride** vor juca un rol cheie în această tranziție către criptografia post-cuantică. Aceste scheme vor combina algoritmi tradiționali, precum RSA, cu tehnici post-cuanti. De exemplu, ar putea fi folosite combinații între RSA și algoritmi precum Lattice-based cryptography sau code-based cryptography pentru a oferi un echilibru între compatibilitate și protecție împotriva amenințărilor cuantice. Acest tip de soluție permite o tranziție lină către tehnologiile criptografice viitoare fără a afecta sistemele existente deodată.

În același timp, **standardizarea criptografiei post-cuanti** devine un aspect esențial. NIST (National Institute of Standards and Technology) a început procesul de dezvoltare a algoritmilor care să reziste atacurilor cuantice. În 2022, NIST a inițiat faza finală a competiției pentru stabilirea unor algoritmi de criptografie post-cuanti care să devină standardizate. Alegerea și implementarea acestor standarde va fi crucială pentru protejarea comunicațiilor și datelor globale în următorii ani.

Prin aceste eforturi, viitorul criptografiei va oferi protecție nu doar împotriva atacurilor clasice, ci și împotriva amenințărilor pe care computerele cuantice le aduc. RSA va fi treptat înlocuit cu scheme mai sigure și mai rezistente, permițând astfel o tranziție controlată către un sistem global mai sigur.

*8.1 Exemple de cod și demonstrație*

Un exemplu simplu în Python pentru generarea cheilor RSA și pentru operațiile de criptare/decriptare utilizează biblioteca pycryptodome, care oferă implementări eficiente și sigure ale algoritmului RSA:

Python

O imagine care conține text, captură de ecran, Font

Descriere generată automat

**Explicație:**

* **Generarea cheilor RSA**: Funcția RSA.generate(2048) generează o pereche de chei RSA cu o lungime de 2048 biți. Aceasta include cheia privată și cheia publică. Cheia publică este utilizată pentru operațiile de criptare, în timp ce cheia privată este folosită pentru decriptare.
* **Criptarea**: Folosim **PKCS1\_OAEP** pentru a efectua criptarea. PKCS1\_OAEP este un algoritm de padding (umplere) folosit pentru a preveni atacurile de text-criptat ales.
* **Decriptarea**: Procesul invers al criptării, utilizând tot PKCS1\_OAEP pentru decriptarea mesajului. Cheia privată este importată și utilizată pentru a descifra mesajul.

*8.2 Experimente cu mărimea cheilor RSA*

Mărimea cheilor RSA este o caracteristică esențială care influențează atât securitatea, cât și performanța operațiilor de criptare și decriptare.

* **1024 biți**: O mărime a cheii RSA mai mică, care a fost folosită în trecut pentru criptare, dar care este considerată nesigură astăzi. Numerele de 1024 biți sunt relativ ușor de factorizat utilizând metodele algoritmului de factorizare GNFS (General Number Field Sieve). Acesta poate fi folosit pentru atacuri de tip brute-force.
* **2048 biți**: Este standardul actual pentru securitate acceptată în majoritatea aplicațiilor de criptare. Acest tip de cheie este suficient de sigur pentru protecția datelor sensibile din prezent și este folosit frecvent în implementările moderne.
* **4096 biți**: O mărime mai mare a cheii, utilizată pentru a asigura o protecție suplimentară, dar cu un cost de performanță semnificativ mai mare. Operațiile de criptare și decriptare devin mult mai lente, iar cheia de 4096 biți oferă o securitate considerabilă împotriva atacurilor clasice.

**Tabel comparativ al timpilor de execuție pentru criptare și decriptare:**

| **Mărimea cheii (biți)** | **Timp criptare (ms)** | **Timp decriptare (ms)** |
| --- | --- | --- |
| 1024 | 5 | 10 |
| 2048 | 20 | 50 |
| 4096 | 100 | 250 |

* **1024 biți**: Operațiile sunt rapide, dar cu o protecție redusă în fața atacurilor.
* **2048 biți**: Balansul optim între securitate și performanță, utilizat frecvent în prezent.
* **4096 biți**: Mai sigur, dar mult mai lent, ideal pentru aplicații care necesită un nivel ridicat de securitate.

*8.3 Analiza securității RSA*

Securitatea RSA depinde de lungimea cheii și de metoda utilizată pentru protecția datelor.

* **Testarea cu unelte de factorizare**:
  + **Numere mici (<512 biți)**: Pot fi factorizate cu ușurință utilizând algoritmi precum GNFS. Un număr cu o lungime de până la 512 biți este considerat nesigur pentru utilizare pe termen lung.
  + **Numere mari (>2048 biți)**: RSA rămâne sigur în fața atacurilor de factorizare, datorită lungimii mari a cheilor.
* **Atacuri practice**:
  + **Atacuri brute-force**: Este extrem de dificil să se obțină cheia privată prin brute-force datorită dimensiunii mari a cheii și complexității operațiunilor matematice.
  + **Erori în implementare**: Una dintre cele mai mari vulnerabilități ale RSA este faptul că erorile de implementare pot dezvălui cheia privată. Aceasta subliniază importanța codării precise și a revizuirii codului criptografic pentru a preveni scurgerile de informații.

Prin urmare, este crucial să se aleagă o mărime adecvată a cheii RSA și să se utilizeze scheme de protecție solide pentru a garanta securitatea datelor în fața atacurilor moderne.

9. Bibliografie

<https://ro.wikipedia.org/wiki/RSA>

[https://chatgpt.com](https://chatgpt.com/?oai-dm=1)

<https://www.geeksforgeeks.org/rsa-algorithm-cryptography/>

<https://www.flt-info.eu/course/isec/>

https://www.csoonline.com/article/3562701/chinese-researchers-break-rsa-encryption-with-a-quantum-computer.html