SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 55

ASIMETRIČNI KRIPTOSUSTAV NTRU

Iva Malović

Zagreb, lipanj 2010.

Mentor rada:

Doc. dr. sc. Marin Golub

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA
POVJERENSTVO ZA DIPLOMSKI ISPIT

Zagreb, 1. ožujka 2010.

DIPLOMSKI ZADATAK br. 55

Pristupnik: **Iva Malović**
Studij: Računarstvo
Profil: Računarska znanost

Zadatak: **Asimetrični kriptosustav NTRU**

Opis zadatka:

Obrazložiti potrebu za novim asimetričnim algoritmom kriptiranja. Opisati osnovnu ideju NTRU kriptosustava (naziv algoritma dolazi od engl. N-th degree TRUncated polynomial ring) i navesti osnovni algoritam. Razmotriti načine poboljšanja NTRU algoritma i opisati njegovu poboljšanu inačicu. Navesti načine na koje se obavlja kriptoanaliza NTRU kriptosustava. Načiniti programski sustav koji omogućuje stvaranje ključeva te kriptiranje i dekriptiranje NTRU algoritmom. Usporediti novu i osnovnu inačicu algoritma s RSA kriptosustavom i pritom posebnu pažnju posvetiti brzini stvaranja ključeva te brzini kriptiranja.

Zadatak uručen pristupniku: 5. ožujka 2010.
Rok za predaju rada: 18. lipnja 2010.

Mentor:

Doc.dr. sc. Marin Golub

Djelovođa:

Doc.dr.sc. Domagoj Jakobović

Predsjednik povjerenstva za diplomski ispit:

Prof.dr.sc. Siniša Srbljić

Sažetak

Danas korišteni asimetrični kriptosustavi se temelje na teškoći problema faktorizacije i diskretnog logaritma. S dolaskom kvantnih računala ti problemi će postati rješivi. Novi kandidati za teške probleme su problemi rešetki. NTRU je jedan od novijih kriptosustava koji se temelji na rešetkama. U ovom radu prezentiran je NTRU kriptosustav i njegova implementacija te usporedba sa široko korištenim RSA kriptosustavom.

Ključne riječi: NTRU, asimetrični kriptosustav, rešetke

Abstract

Current asymmetrical cryptosystems are based on factorization problem and discrete logarithm problem. With quantum computers, these problems will become solvable. New candidates for hard problem in asymmetrical cryptography are lattice problems. NTRU is one of the new cryptosystems based on lattice problems. This paper presents the NTRU cryptosystem and its implementation compared to widely used RSA cryptosystem.

Keywords: NTRU, asymmetric cryptosystem, lattice

Sadržaj

Popis slika

Popis tablica

1. Uvod

Shor je 1994. objavio članak [1] o kvantnim algoritmima polinomne složenosti za faktoriziranje velikih brojeva i određivanje diskretnog logaritma. U studenom 2009. NIST je objavio članak [2] o prvom programabilnom kvantnom procesoru. Kvantna računala će postati stvarnost puno prije nego se to predviđalo ranije, a time će sva današnja asimetrična kriptografija postati nesigurna (RSA, Diffie-Hellman, ElGamal, ECC). Ipak, kvantna računala ne znače kraj kriptografije i sigurnosti Interneta, već samo potrebu za upotrebom novih kriptografskih algoritama.

U ovom radu opisan je NTRU kriptosustav – do sada najbolji kandidat za asimetrični kriptosustav u vrijeme kvantnih računala.

U drugom poglavlju opisani su osnovni pojmovi vezani uz kriptografiju. U trećem poglavlju opisana je matematička osnova na kojoj se temelji NTRU kriptosustav. Slijedi pregled aktualnih asimetričnih kriptosustava, te uvod u kvantna računala i njihov utjecaj na današnje asimetrične kriptosustave. Zatim slijedi opis NTRU kriptosustava i digitalnog potpisa te kriptoanaliza NTRU-a. Na kraju su opisani detalji o implementaciji NTRU kriptosustava i usporedba s RSA kriptosustavom.

2. Uvod u kriptografiju i kriptoanalizu

Prilikom komunikacije s nekom osobom, želimo biti sigurni da komuniciramo upravo s tom osobom i da komunikaciju ne prisluškuje (ne razumije) treća osoba. Osim toga, želimo biti sigurni da se podaci koje primimo ili šaljemo nisu u međuvremenu promijenili zbog uplitanja uljeza. Te zahtjeve osigurava kriptografija.

Kriptografija (grč. *kryptós* – skriven + *gráfo* – pisati) [3] je znanstvena disciplina koja primjenom matematike istražuje načine pretvorbe informacije iz razumljivog oblika u nečitljiv oblik kako bi se zaštitila privatnost informacije, njena vjerodostojnost i porijeklo. Drugim riječima, kriptografija služi za zadovoljavanje sljedećih *sigurnosnih zahtjeva* [4]:

1. Tajnost, povjerljivost (engl. *confidentiality*) – informacije smiju biti dostupne samo ovlaštenim korisnicima
2. Integritet, besprijekornost (engl. *integrity*) – informacije mogu mijenjati samo za to ovlašteni korisnici
3. Autentifikacija (engl. *authenticity*) – ovlašteni se korisnici moraju moći jednoznačno prepoznati
4. Neporecivost (engl. *non-repudiation*) – nemogućnost opovrgavanja

Kriptoanaliza (grč. *kryptós* – skriven + *analýein* – razmrsiti) [5] je znanstvena disciplina koja se bavi proučavanjem postupaka za čitanje skrivenih poruka bez poznavanja ključa. To ne znači da kriptoanaliza narušava ljudsku privatnost i poništava kriptografiju – naprotiv, kriptografija i kriptoanaliza neizostavno su povezane i nadopunjuju se. Svrha kriptoanalize je analiza i pronalaženje propusta u kriptografskim algoritmima, jer bez analize, sigurnost algoritama je samo pretpostavka. Dakle, kriptoanaliza je znanost koja kriptografiju čini sigurnom.

2.1. Osnovni kriptografski pojmovi

Jasni tekst (čisti tekst) je izvorni, razumljiv oblik informacije.

Kriptografski algoritam je slijed precizno definiranih koraka kojima se iz čistog teksta dobiva kriptirani tekst ili obratno.

Ključ je ulazni parametar kriptografskog algoritma temeljem kojeg se čisti tekst prevodi u kriptirani tekst, ili obratno, kriptirani tekst u čisti tekst.

Kriptirani tekst (šifrat) je reprezentacija čistog teksta dobivena kriptografskim algoritmom, koja se ne može pročitati bez poznavanja odgovarajućeg ključa.

Postupkom *kriptiranja* čisti tekst se upotrebom kriptografskog algoritma prevodi u kriptirani tekst. Obrnuti postupak prevođenja kriptiranog teksta u čisti tekst naziva se *dekriptiranje*.

Kriptosustav se sastoji od kriptografskog algoritma (obično jedan za kriptiranje i jedan za dekriptiranje) te skupa svih mogućih čistih tekstova, kriptiranih tekstova i ključeva. [4][6]

2.2. Kriptografski primitivi

Kriptografski primitivi [7] su osnovni građevni blokovi sigurnosnih sustava. Kriptografski primitivi sami po sebi ne osiguravaju sve sigurnosne zahtjeve, već se to ostvaruje njihovim kombiniranjem.

2.2.1. Simetrični kriptosustavi

Simetrični kriptosustavi koriste isti ključ za kriptiranje i za dekriptiranje koji se još naziva *tajni ključ*, ili *sjednički ključ*. U simetričnim algoritmima kriptiranja najčešće se koristi logička operacija ekskluzivno ili (XOR).

Najpoznatiji simetrični algoritmi kriptiranja su DES (*Data Encryption Standard*), 3DES (*Triple DES*), AES (*Advanced Encryption Standard*), IDEA, RC4.

Simetrični kriptosustavi su veoma brzi, ali glavna mana im je potreban broj ključeva i njihova razmjena. Za komuniciranje n sudionika potrebno je $\frac{n(n-1)}{2}$ ključeva. Problem sigurne razmjene ključeva se rješava upotrebom asimetrične kriptografije. [4]

2.2.2. Asimetrični kriptosustavi

Asimetrični kriptosustavi koriste par ključeva, jedan za kriptiranje, drugi za dekriptiranje. Asimetrični algoritmi kriptiranja su daleko sporiji od simetričnih algoritama i imaju dulji ključ. Iz tog razloga, simetrični algoritmi se koriste za kriptiranje veće količine podataka, dok se asimetrični algoritmi koriste za razmjenu ključeva koji se koriste u simetričnim algoritmima. Osim razmjene ključeva, asimetrični kriptosustavi se koriste u digitalnim potpisima za osiguravanje autentičnosti. [4]

Više o asimetričnoj kriptografiji nalazi se u poglavlju 4.

2.2.3. Sažetak poruke

Sažetak (engl. *hash*) poruke je niz bitova fiksne duljine koji se dobiva pomoću funkcija sažimanja.

Funkcije sažimanja moraju imati sljedeća svojstva:

- Funkcija sažimanja mora biti jednosmjerna – iz sažetka mora biti nemoguće dobiti izvornu poruku;
- Svaki par različitih poruka mora se preslikati u različite sažetke, čak i ako se poruke razlikuju u samo jednom bitu;
- Svaki put kad se ista poruka pusti kroz istu funkciju sažimanja, rezultati moraju biti identični;
- Duljina sažetka je određena funkcijom sažimanja i ne mijenja se s duljinom poruke.

Funkcije sažimanja se koriste za osiguravanje integriteta poruke, a u kombinaciji s asimetričnim algoritmima koriste se za osiguravanje porijekla poruke. [4]

2.2.4. Digitalni potpis

Algoritmima kriptiranja osigurava se jedino tajnost poruke. Naime, Bob ne može biti siguran da mu je upravo Alice poslala poruku. Svatko ima pristup algoritmima za kriptiranje, pa se može lažno predstaviti kao Alice. Digitalnim potpisom se rješava taj problem.

Pri izradi digitalnog potpisa prvo se mora izračunati sažetak poruke te se zatim poruka potpisuje uz pomoć privatnog ključa. Time se točno zna tko je poslao poruku i je li poruka u međuvremenu promijenjena.

Digitalni potpis se formalno definira na sljedeći način:

$$Digitalni\ potpis = E(H(m), S_A), \tag{2.1}$$

gdje je m poruka, $H(m)$ sažetak poruke m, S_A privatni ključ potpisnika i E asimetrični algoritam kriptiranja. [4]

3. Matematički preduvjeti

3.1. Kongruencije [8]

Definicija 3.1 (Kongruentnost) Ako cijeli broj $m \neq 0$ dijeli razliku $a - b$ (pišemo $m|a - b$), onda kažemo da je a kongruentan b modulo m i pišemo $a \equiv b \ (mod\ m)$. U protivnom, kažemo da a nije kongruentan b modulo m i pišemo $a \not\equiv b \ (mod\ m)$.

Neka od važnijih svojstava kongruencija su:

i. Ako je $a \equiv b \ (mod\ m)$, $c \equiv d \ (mod\ m)$, onda je $a + c \equiv b + d \ (mod\ m)$, $a - c \equiv b - d \ (mod\ m)$, $ac \equiv bd \ (mod\ m)$.
ii. Ako je $a \equiv b \ (mod\ m)$ i $d|m$, onda je $a \equiv b \ (mod\ d)$.
iii. Ako je $a \equiv b \ (mod\ m)$, onda je $ac \equiv bc \ (mod\ mc)$ za svaki $c \neq 0$.

Definicija 3.2 (Eulerova funkcija) Eulerova funkcija, $\varphi(m)$ je broj brojeva u nizu $1,2,\dots,m$ koji su relativno prosti sa m.

Primjer:

$$\varphi(12) = 7 \ jer\ je\ nzd(12,x) = 1 \ za\ x = \{1,5,7,8,9,10,11\}$$

Teorem 3.1 (Eulerov teorem) Ako je $nzd(a,m) = 1$, onda je $a^{\varphi(m)} \equiv 1 \ (mod\ m)$.

Teorem 3.2 (Mali Fermatov teorem) Neka je p prost broj. Ako $p \nmid a$, onda je $a^{p-1} \equiv 1 \ (mod\ p)$.

3.2. Algebarske strukture

3.2.1. Grupa

Definicija 3.3 (Polugrupa) Skup X s asocijativnom binarnom operacijom zove se polugrupa.

Binarna operacija $\circ$ je asocijativna ako vrijedi:

$$x \circ (y \circ z) = (x \circ y) \circ z, za\ \forall x,y,z \in X. \tag{3.1}$$

Definicija 3.4 (Grupa) Grupa je neprazan skup G zajedno s binarnom operacijom $\circ$, kojom dvama elementima x i y iz G pridružujemo element $x \circ y$ u G, tako da vrijedi:

i. asocijativnost: $x \circ (y \circ z) = (x \circ y) \circ z$, za sve $x, y, z \in G$
ii. postojanje neutralnog elementa: neutralni element je neki $e \in G$ sa svojstvom da za sve $x \in G$ vrijedi $x \circ e = e \circ x = x$
iii. postojanje inverza: za svaki $x \in G$ postoji element $x^{-1} \in G$ takav da je $x \circ x^{-1} = x^{-1} \circ x = e$

Za grupu kažemo da je *komutativna* ili *abelova* ako vrijedi $x \circ y = y \circ x$, za sve $x, y \in G$.

Ako je binarna operacija u grupi $G = G(X,\circ)$ množenje $(\cdot)$, grupa G se naziva *multiplikativna* grupa i označava se s $G(X, \cdot)$. Grupa G je *aditivna* grupa ako je korištena binarna operacija operacija zbrajanja i označava se $G(X, +)$. [8]

Definicija 3.5 (Homomorfizam) Neka su G i H grupe. Preslikavanje $\phi: G \to H$ naziva se homomorfizam ako vrijedi:

$$\phi(xy) = \phi(x)\phi(y) \; za \; svaki \; x, y \in G. \tag{3.2}$$

Homomorfizam koji je ujedno i bijekcija[1] naziva se *izomorfizam*. [9]

3.2.2. Prsten

Definicija 3.6 (Prsten) Prsten je skup R zajedno s dvije binarne operacije $+$ i $\cdot$ za koje vrijedi:

i. $(R, +)$ je abelova grupa,
ii. $(R, \cdot)$ je polugrupa,
iii. $x \cdot (y + z) = x \cdot y + x \cdot z$ i $(x + y) \cdot z = x \cdot z + y \cdot z$, za $\forall x, y, z \in R$ (distributivnost $\cdot$ s obzirom na $+$).

Definicija 3.7 (Ideal) Neprazan skup I prstena R naziva se ideal ako vrijedi

i. $a - b \in I$, za svaki $a, b \in I$
ii. $ra \in I$ i $ar \in I$, za svaki $a \in I, r \in R$.

Svaki ideal je podprsten, ali svaki podprsten nije nužno ideal.

[1] Za funkciju iz skupa *X* u skup *Y* kažemo da je bijektivna ako za svaki *y* u *Y* postoji točno jedan *x* u *X* takav da je *f*(*x*) = *y*. Bijekcija se još naziva 1-1 preslikavanje (jedan na jedan preslikavanje)

Definicija 3.8 (Kvocijentni prsten) Neka je I ideal prstena R. Definiramo relaciju $\sim$ na R sa $a \sim b$ akko $a - b \in I$. Lako je provjeriti da je relacija $\sim$ relacija kongruentnosti. Klasa ekvivalencije elementa $a \in R$ je

$$\bar{a} = \{b \in R | a - b \in I\} = \{a + r | r \in I\} = a + I. \tag{3.3}$$

Klasa ekvivalencije se ponekad piše kao $a \, mod \, I$ i naziva se rezidualna klasa od a modulo I.

Skup svih klasa ekvivalencija $\bar{a}$ naziva se kvocijentni prsten i označava sa

$$R/I = \{a + I | a \in R\}. \tag{3.4}$$

Na skupu R/I definiraju se operacije zbrajanja i množenja:

$$(a + I) + (b + I) = (a + b) + I, \tag{3.5}$$

$$(a + I)(b + I) = ab + I. \tag{3.6}$$

Neutralni element za zbrajanje je klasa ekvivalencije $0 + I = I$. Ako R ima jedinicu, tada R/I ima neutralni element za množenje $1 + I$. [9]

Definicija 3.9 (Prsten polinoma) Neka je $(R, +, \cdot)$ neki prsten. Polinom u varijabli x nad R je svaki izraz (formalna suma) oblika

$$p(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0 , \tag{3.7}$$

gdje su $a_0, a_1, \ldots, a_n \in R, n \in N_0$.

Skup svih polinoma nad R označavamo s $R[x]$. U $R[x]$ se uvode operacije zbrajanja i množenja. Ako je $p(x)$ zadan s relacijom (3.7) i $q(x) = b_m x^m + b_{m-1} x^{m-1} + \cdots + b_1 x + b_0$, onda je zbroj polinoma p i q jednak

$$p(x) + q(x) = \sum_{k=0}^{\max(m,n)} (a_k + b_k) x^k, uz \; a_k = 0 \; za \; k > n \; i \; b_k = 0 \; za \; k > m, \tag{3.8}$$

dok je umnožak polinoma p i q jednak

$$p(x) \cdot q(x) = \sum_{k=0}^{m+n} \left(\sum_{i+j=k} a_i b_j \right) x^k \tag{3.9}$$

Skup $R[x]$ uz ove dvije operacije čini prsten. [8]

3.2.3. Kvocijentni prsten polinoma

Za potrebe NTRU kriptosustava definira se kvocijentni prsten polinoma:

$$R = Z[X]/(X^N - 1). \tag{3.10}$$

Elementi tog prstena su polinomi stupnja manjeg od N s cjelobrojnim koeficijentima.

Zbrajanje elemenata unutar R se izvodi na jednak način kao i kod običnih prstena polinoma, zbrajaju se koeficijenti na istim pozicijama.

Množenje elemenata u kvocijentnom prstenu polinoma se naziva *konvolucijsko množenje* (ili cirkularno množenje), oznake $\circledast$ i definira se na slijedeći način:

$$\begin{gathered} r(x) = p(x) \circledast q(x) = r_0 + r_1 x + r_2 x^2 + \cdots + r_{N-1} x^{N-1} + r_N x^N, uz \\ r_k = a_0 b_k + a_1 b_{k-1} + \cdots + a_k b_0 + a_{k+1} b_N + a_{k+2} b_{N-1} + \cdots + a_N b_{k+1} \end{gathered} \tag{3.11}$$

Pri tome je $N = max\{n, m\}$, gdje su n i m stupnjevi polinoma p odnosno q.

Konvolucijsko množenje je zapravo standardno množenje polinoma definirano u relaciji (3.9) uz redukciju potencija od x modulo N.

Definicija 3.10 (Multiplikativni inverz polinoma) Inverz polinoma a modulo q iz prstena polinoma je polinom A za koji vrijedi

$$aA \equiv 1 (mod\ q). \tag{3.12}$$

3.2.4. Vektorski prostor

Definicija 3.11 (Polje) Polje je skup F zajedno s dvije binarne operacije $+$ i $\cdot$, koje bilo kojim dvama elementima $\lambda, \mu \in F$ pridružuje $\lambda + \mu \in F$ i $\lambda \cdot \mu \in F$, tako da vrijedi:

i. $(F, +)$ je aditivna grupa
ii. skup $F^* \coloneqq F \backslash \{0\}$ je komutativna grupa s obzirom na množenje
iii. operacije zbrajanja i množenja su usklađene zakonom distribucije: $\lambda(\mu + v) = \lambda\mu + \lambda v$.

Definicija 3.12 (Vektorski prostor) Vektorski prostor nad zadanim poljem F je skup X zajedno s dvije operacije:

i. operacijom zbrajanja $(+)$ u X
ii. operacijom $\cdot$ kojom bilo kojem skalaru $\lambda \in F$ i elementu (vektoru) $x \in X$ pridružujemo $\lambda x \in X$ (ovo množenje nije isto što i množenje u polju F),

tako da vrijedi:

a. $(X, +)$ je aditivna grupa, elemente od X zovemo vektorima
b. usklađenost operacija množenja skalara u polju F i množenje skalara s vektorima u X: $\lambda(\mu x) = (\lambda\mu)x, 1x = x$, za sve $\lambda, \mu \in F\ i\ x \in X$
c. zakoni distribucije: $(\lambda + \mu)x = \lambda x + \mu x, \lambda(x + y) = \lambda x + \lambda y$, za sve $\lambda, \mu \in F\ i\ x, y \in X$.

Kada govorimo o vektorskom prostoru $(X, +, \cdot)$, onda se operacija $\cdot$ odnosi na množenje skalara iz F s vektorima iz X.

U vektorskom prostoru X uvodimo *linearnu kombinaciju* vektora $x_1, \dots, x_k \in X$ kao izraz $\lambda_1 x_1 + \cdots + \lambda_k x_k \in X$.

Za vektore $x_1, \dots, x_k \in X$ kažemo da su *linearno nezavisni* ako vrijedi da

$$\lambda_1 x_1 + \cdots + \lambda_k x_k = 0 \quad \Rightarrow \quad \lambda_1 = \cdots = \lambda_k = 0. \tag{3.13}$$

Definicija 3.13 (Linearna ljuska) Skup svih linearnih kombinacija vektora $x_1, \dots, x_k \in X$ naziva se linearna ljuska od X ili prostor razapet vektorima $x_1, \dots, x_k$ (engl. *span*). [10]

3.3. Rešetke

Definicija 3.14 (Rešetka) Neka je R^m m-dimenzijski euklidski prostor. Rešetka u R^m je skup svih mogućih kombinacija n linearno nezavisnih vektora $b_1, \dots, b_n$ iz R^m $(m \geq n)$:

$$L = L(b_1, \dots, b_n) = \left\{ \sum_{i=1}^{n} x_i b_i \ \middle| x_i \in Z\ za\ \forall i \right\}. \tag{3.14}$$

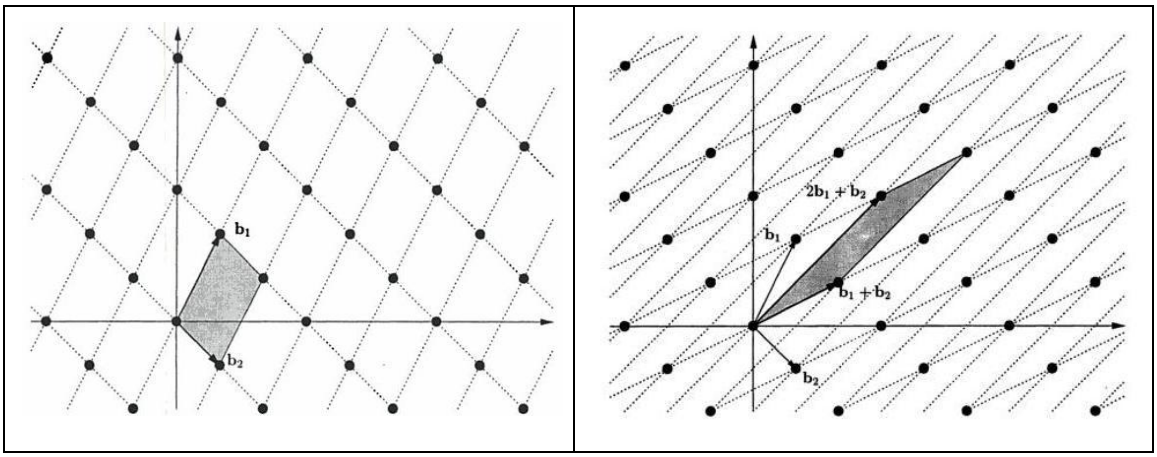
Vrijednosti n i m nazivaju se rang i dimenzija rešetke respektivno. Kažemo da je rešetka $L(B)$ potpuno dimenzijska kad je $n = m$. Slijed vektora $b_1, \dots, b_n$ naziva se bazom rešetke i obično je predstavljen kao matrica gdje su vektori $b_1, \dots, b_n$ stupci: $B = [b_1, \dots, b_n] \in R^{m \times n}$. [12]

Upotrebljavajući matričnu notaciju rešetka se može zapisati kao

$$L(B) = \{Bx : |x \in Z^n\ za\ \forall x\}. \tag{3.15}$$

Duljina baze rešetke je duljina najduljeg vektora baze, $\max_{1 \leq i \leq n} \|b_i\|$.

Na slici 3.1 (lijevo) prikazana je dvodimenzionalna rešetka s vektorima baze $b_1 = [1 \quad 2]$, $b_2 = [1 \quad -1]$. Ista rešetka može imati više različitih baza. Na primjer baza rešetke na slici 3.1 (lijevo) može biti $b_1' = b_1 + b_2 = [2\ 1]$, $b_2' = 2b_1 + b_2 = [3\ 3]$. Mreže koje tvore vektori (b_1, b_2) i (b_1', b_2') su različite, ali presjecišta su na istim koordinatama kao što se vidi na slici 3.1. [11]



Slika 3.1 Primjer dvodimenzionalne rešetke s različitim bazama: lijevo – baza: $b_1 = [1 \quad 2],\ b_2 = [1 \quad -1]$; desno – baza: $b_1' = [2\ \ 1],\ b_2' = [3\ \ 3]$ [11]

Definicija 3.15 (Linearna ljuska rešetke) [12] Linearna ljuska rešetke je prostor razapet vektorima rešetke i ne ovi o specifičnoj bazi:

$$span\big(L(B)\big) = span(B) = \{Bx \mid x \in R^n\}. \tag{3.16}$$

Definicija 3.16 (Ekvivalentnost baza rešetke) Dvije baze $B, B' \in R^{m\times n}$ su ekvivalentne ako i samo ako postoji unimodularna matrica[2] $U \in Z^{n\times n}$ takva da je $B' = BU$.

Definicija 3.17 (Temeljni paralelepiped) Za zadanu bazu B rešetke $L(B)$, temeljni paralelepiped (engl. *fundamental paralelepiped*) definira se kao

$$P(B) = \{Bx : x \in [0,1)^n\}. \tag{3.17}$$

Rešetka ima različite paralelepipede ovisno o specifičnoj bazi.

Definicija 3.18 (Determinanta rešetke) Neka je $B \in R^{m\times n}$ baza rešetke. Determinanta rešetke je definirana kao n-dimenzijski volumen fundamentalnog paralelepipeda koji odgovara bazi B (osjenčani dio na slici 3.1):

$$det\big(L(B)\big) = vol\big(P(B)\big). \tag{3.18}$$

Determinanta je neovisna o bazi koja je generirala rešetku.

Kada je $B \in R^{n\times n}$ nesingularna[3] kvadratna matrica, determinanta rešetke je jednaka

$$det\big(L(B)\big) = |det(B)|. \tag{3.19}$$

Drugi način računanja determinante rešetke je preko formule

[2] Unimodularna matrica je kvadratna matrica s determinantom jednakom +1 ili -1.
[3] Nesingularna matrica je kvadratna matrica kojoj je determinanta različita od nule.

$$det\big(L(B)\big) = \sqrt{det\,(B^T B)}\,. \tag{3.20}$$

Definicija 3.19 (i-ti sukcesivni minimum) Neka je L rešetka ranga n. Za $i = 1, \dots, n$, i-ti sukcesivni minimum je

$$\lambda_i(L(B)) = inf\left\{r : dim\left(span\left(L \cap \bar{\beta}(0,r)\right)\right) \geq i\right\}, \tag{3.21}$$

gdje je $\bar{\beta}(0,r) = \{x \in R^m : \|x\| \leq r\}$ zatvorena kugla radijusa r oko ishodišta.

Definicija 3.20 (Prvi sukcesivni minimum) Prvi sukcesivni minimum je minimalna udaljenost između bilo koje različite točke rešetke i jednaka je duljini najkraćeg ne-nul vektora rešetke:

$$\lambda_1(L(B)) = min\{\|x\|_2 : x \in L \setminus \{0\}\}. \tag{3.22}$$

Dakle, prvi sukcesivni minimum, λ_1 označava radijus najmanje kugle centrirane u ishodištu koja sadrži ne-nul vektor rešetke. Jednako tome, λ_2 je radijus najmanje kugle koja sadrži dva linearno nezavisna ne-nul vektora rešetke.

Teorem 3.3 (Hermitov teorem) Svaka rešetka L dimenzije n sadrži ne-nul vektor $v \in L$ za koji vrijedi

$$\|v\| \leq \sqrt{n}\; det(L)^{1/n}. \tag{3.23}$$

Izraz $\sqrt{n}\; det(L)^{1/n}$ zapravo predstavlja gornju granicu prvog minimuma rešetke λ_1.

Za zadanu dimenziju n, Hermitova konstanta γ_n je najmanja vrijednost takva da svaka rešetka L dimenzije n sadrži ne-nul vektor $v \in L$ za koji vrijedi

$$\|v\|^2 \leq \gamma_n det(L)^{2/n}. \tag{3.24}$$

Za velike n poznato je da γ_n zadovoljava

$$\frac{n}{2\pi e} \leq \gamma_n \leq \frac{n}{\pi e}. \tag{3.25}$$

Teorem 3.4 (Minkowskov drugi teorem) Za svaku rešetku $L(B)$ ranga n, sukcesivni minimumi (u 2-normi) $\lambda_1, \dots, \lambda_n$ zadovoljavaju [11]

$$\left(\prod_{i=1}^{n} \lambda_i\right)^{\frac{1}{n}} < \sqrt{n}\; det(\mathrm{B})^{\frac{1}{n}}\,. \tag{3.26}$$

4. Asimetrični kriptosustavi

U asimetričnim kriptosustavima se dakle, rabe dva ključa: privatni i javni. Javni ključ je dostupan svima i njime se kriptira poruka, dok je privatni ključ dostupan samo vlasniku koji njime dekriptira poruku. Iz javnog ključa ne smije biti moguće izračunati privatni ključ. Kako je javni ključ poznat svima, može se dogoditi da se netko lažno predstavi te su za osiguravanje autentičnosti sugovornika u komunikaciji potrebne dodatne tehnike.

4.1. Teško rješivi problemi

U asimetričnim kriptosustavima upotrebljavaju se tzv. jednosmjerne funkcije s tajnim vratima (engl. *one-way trapdoor function*). Svojstvo tih funkcija je da su jednostavne za računanje u jednom smjeru, a vrlo teške za invertiranje bez dodatnih informacija. [14]

4.1.1. Faktorizacija

Definicija 4.1 (Problem faktorizacije) Za zadan N za koji vrijedi $N = pq$ gdje su p i q prosti brojevi, nađi p i q.

Za velike p i q problem faktorizacije se smatra NP teškim[4]. No, ne postoji dokaz da je problem faktorizacije uistinu NP-težak. Najbrži poznati algoritam (*number sieve*) ima složenost $O\left(N^{1/3}\right)$. [15]

4.1.2. Diskretni logaritam

Definicija 4.2 (Problem diskretnog logaritma) Za dani N i neki broj x, nađi najmanji broj r takav da vrijedi $x^r \equiv 1(mod\ N)$.

Ovaj problem se također smatra NP-teškim. [15]

[4] Problem je NP-težak ako ne postoji algoritam polinomne složenosti za njegovo rješavanje. Instance problema koji je NP-težak se općenito mogu lakše riješiti što najviše zabrinjava kad je riječ o asimetričnim kriptosustavima

4.1.3. Problem naprtnjače

Problem naprtnjače (engl. *knapsack problem*) je kombinatorni problem optimizacije.

Definicija 4.3 (Problem naprtnjače) Za zadani skup elemenata s težinama i vrijednostima, odredi broj elemenata od svake vrste elemenata koji se trebaju uključiti u traženi skup tako da je ukupna težina manja od zadanog limita i da je ukupna vrijednost (suma) što je moguće veća.

Problem se može preformulirati u svakodnevnu upotrebu: Recimo da postoji 10 komada hrane od kojih svaki komad ima određenu nutritivnu vrijednost i svoju težinu. Problem naprtnjače je odabrati podskup hrane takav da se ne prijeđe zadani limit u ukupnoj težini, a da je nutritivna vrijednost što veća.

Problem naprtnjače je NP-potpun problem, dakle još je teži od problema faktorizacije i diskretnog logaritma. [16]

4.1.4. Problemi rešetke

Postoji nekoliko problema rešetki za koje se vjeruje da su NP teški:

1. Problem najkraćeg vektora (engl. *Shortest Vector Problem* – SVP)
2. Problem najbližeg vektora (engl. *Closest Vector Problem* – CVP)

Algoritmi za njihovo rješavanje nazivaju se algoritmi redukcije rešetke. Formalna definicija redukcije rešetke bila bi nalaženje najkraće baze rešetke. Najbolji algoritmi za rješavanje ovog problema su ili eksponencijalne složenosti ili daju loše aproksimacijske rezultate. [11]

Definicija 4.4 (Problem najkraćeg vektora) Uz danu bazu rešetke $B \in Z^{m \times n}$, nađi ne-nul vektor rešetke Bx (uz $x \in Z^n \backslash \{0\}$) takav da $\|Bx\| \leq \|By\|$ za svaki $y \in Z^n \backslash \{0\}$.

SVP problem je zapravo traženje vektora rešetke koji postiže prvi sukcesivni minimum. Ne postoji algoritam polinomne složenosti za njegovo rješavanje. Postoji aproksimacijski algoritam (LLL algoritam [17]), koji u polinomnom vremenu nalazi vektor rešetke čija je duljina najviše γ puta veća od duljine najkraćeg vektora rešetke. Faktor γ se naziva aproksimacijski faktor. Najbolji aproksimacijski faktor jednak je $\gamma = \left(\frac{2}{\sqrt{3}}\right)^n$. Za bolje aproksimacije upotrebljava se BKZ [18] (Block Korkin-Zoloratev) algoritam od Schnorra i Euchnera.

Gaussova heuristika kaže da je duljina najkraćeg ne-nul vektora obično približno jednaka

$$\lambda_1(L) \approx \sqrt{\frac{dim(L)}{2\pi e}}\, det(L)^{\frac{1}{dim(L)}}, \tag{4.1}$$

gdje je $dim(L)$ dimenzija rešetke L. [11]

Definicija 4.5 (Problem najbližeg vektora) Uz danu bazu rešetke $B \in Z^{m \times n}$ i vektor $t \in Z^m$, nađi vektor rešetke Bx najbliži vektoru t. Drugim riječima, nađi vektor $x \in Z^n$ takav da $\|Bx - t\| \leq \|By - t\|$ za svaki $y \in Z^n$.

Za CVP problem postoji aproksimacijski algoritam polinomne složenosti (Babai-jeva tehnika [19]). Aproksimacijski faktor jednak je $2\left(\frac{2}{\sqrt{3}}\right)^n$, gdje je n rang rešetke, što je dosta loše rješenje. [11]

4.2. Pregled asimetričnih kriptosustava

Tablica 4.1 Pregled asimetričnih kriptosustava

Asimetrični algoritam	**NP-težak problem**
RSA	problem faktoriziranja
Rabin	problem faktoriziranja
Blum-Goldwasser	problem faktoriziranja
Diffie-Hellman	diskretni logaritam
ECC	diskretni logaritam
ElGamal	diskretni logaritam
DSS	diskretni logaritam
LUC	diskretni logaritam
Rivest-Chor	problem naprtnjače
Merkle-Hellman	problem naprtnjače
McEliece	dekodiranje linearnog koda

U tablici 4.1 dan je pregled najpoznatijih asimetričnih algoritama i problema na kojima se zasnivaju. Od navedenih algoritama najšire korišteni asimetrični algoritmi su ECC[5] [6] te RSA.

[5] ECC (engl. Eliptic Curve Cryptography) – kriptografija zasnovana na eliptičkim krivuljama.

4.2.1. RSA kriptosustav

RSA kriptosustav, nazvan prema svojim tvorcima (R. Rivest, A. Shamir, L. Adleman), osmišljen je 1977. godine i zasniva se na teškosti faktoriziranja velikih brojeva.

Javni ključ RSA kriptosustava je par brojeva n i e, $K_E = (e, n)$. Dok je privatni ključ par brojeva n i d, $K_D = (d, n)$. Ključevi se generiraju na sljedeći način:

1. Odaberu se dva velika prosta broja p i q
2. Izračuna se umnožak $n = p \cdot q$
3. Izračuna se umnožak $\varphi(n) = (p-1) \cdot (q-1)$
4. Odabere se broj $d < \varphi(n)$, koji je relativno prost sa $\varphi(n)$
5. Izračuna se broj $e < \varphi(n)$ tako da vrijedi $e \cdot d \equiv 1\big(mod\ \varphi(n)\big)$

Postupak kriptiranja jednak je $C = E(P, K_E) = P^e\ mod\ n$, a postupak dekriptiranja jednak je $P = D(C, K_D) = C^d\ mod\ n$.

Do privatnog ključa RSA kriptosustava napadač može doći ako uspije faktorizirati broj n, odnosno ako sazna brojeve p i q.

Parametar e naziva se javni eksponent i u većini slučajeva fiksira se na određenu vrijednost (3, 5, 7, ili 65537).

Prilikom generiranja velikih prostih brojeva koristi se algoritam za provjeru prostosti. Najefikasniji algoritam za provjeru prostosti je Miller-Rabinov algoritam. Ako nakon k iteracija algoritma broj prođe test prostosti, broj je prost sa sigurnošću:

$$p = 1 - 2^{-2k} \tag{4.2}$$

Dakle, već nakon samo 5 iteracija Miller-Rabinovog testa prostosti broj je prost sa sigurnošću od 0.999. [4]

5. Kvantna računala i kriptografija

„Svatko tko nije šokiran kvantnom teorijom nije je razumio!“ –
Niels Bohr

Kvantna mehanika razvila se tijekom prvih 30 godina 20. stoljeća kao odgovor na neuspjeh klasične Newtonove mehanike na atomskoj i subatomskoj razini. Prema kvantnoj teoriji, materija je istodobno i val i čestica. Kod subatomskih čestica, poput fotona, to valno-čestično svojstvo dolazi do velikog izražaja.

Prema teoriji kvantne mehanike, foton se može nalaziti na više putova istovremeno te se jednako tome, pobuđeni elektron istovremeno može nalaziti u dvije orbite. To dolazi iz valnog svojstva tvari. Takva stvar je normalnom čovjeku nezamisliva i protivi se intuiciji. Zbog toga je čak i Einstein smatrao da je kvantna teorija pogrešna.

U kvantnoj mehanici koristi se unitarni prostor[6] C^n. Taj prostor je konačne dimenzije i naziva se još Hilbertovim prostorom. Vektor u tom prostoru predstavlja stanje neke čestice nad kojim djeluju unitarni[7] i hermitski[8] linearni operatori. [20]

5.1. Qubit

5.1.1. Diracova notacija

U klasičnim računalima osnovni procesni element je bit koji poprima vrijednosti $\{0,1\}$. U kvantnim računalima osnovni procesni element je *qubit* (engl. *quantum bit*). U qubitu, kvantno logičko stanje 0 odgovara vektoru $\binom{1}{0}$, a kvantna vrijednost 1 odgovara vektoru $\binom{0}{1}$. Standardno se stanje qubita zapisuje Diracovom notacijom [20]:

$$|0\rangle \equiv \binom{1}{0} , |1\rangle \equiv \binom{0}{1}. \tag{5.1}$$

Vektori $\{|0\rangle, |1\rangle\}$ čine bazu Hilbertovog prostora C^2.

Qubit može poprimiti i stanje koje je superpozicija stanja baze:

[6] Vektorski prostor u kojem je definiran skalarni produkt naziva se unitarni prostor.
[7] Operator U se naziva unitarnim ako vrijedi $UU^\dagger = U^\dagger U = I$
[8] Neki linearni operator A naziva se hermitskim ako vrijedi $A = A^\dagger$, gdje je $A^\dagger = (A^T)^*$

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle, \qquad uz\ \alpha, \beta \in C\ , |\alpha|^2 + |\beta|^2 = 1. \tag{5.2}$$

Vrijednosti α, β označavaju amplitude pojedinog stanja od $|\psi\rangle$, a njihovi kvadrati apsolutnih vrijednosti predstavljaju vjerojatnosti pojavljivanja pojedinih stanja prilikom mjerenja qubita. Vjerojatnost da je izmjerena vrijednost od $|\psi\rangle$ jednaka 0 je $|\alpha|^2$, dok je vjerojatnost za vrijednost 1 jednaka $|\beta|^2$. Jednom kada se sustav izmjeri, on ostaje u izmjerenom stanju čime se prethodno stanje uništilo.

Vektori $|0\rangle, |1\rangle, |\psi\rangle$ su stupčani vektori i nazivaju se *ket*. Jednako tome, retčani vektori (*bra* vektori) se definiraju kao

$$\langle 0| \equiv \begin{pmatrix} 1 & 0 \end{pmatrix}, \langle 1| \equiv \begin{pmatrix} 0 & 1 \end{pmatrix}, \langle \psi| \equiv \begin{pmatrix} \bar{\alpha} & \bar{\beta} \end{pmatrix}. \tag{5.3}$$

Pretvorba *ket* vektora $|\psi\rangle$ u njegov dualni *bra* vektor $\langle\psi|$ naziva se Hermitska konjugacija i označava se $\langle\psi| = |\psi\rangle^{\dagger}$.

Skalarni produkt dva vektora $|\psi\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}$ i $|\phi\rangle = \begin{pmatrix} \gamma \\ \delta \end{pmatrix}$ jednak je:

$$\langle\phi|\psi\rangle = \begin{pmatrix} \bar{\gamma} & \bar{\delta} \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \alpha\bar{\gamma} + \beta\bar{\delta}. \tag{5.4}$$

Skalarni produkt $\langle\phi|\psi\rangle$ naziva se *bracket*. Uoči da je $\langle\psi|\psi\rangle = |\alpha|^2 + |\beta|^2 = 1$.

Kvantno stanje niza qubita se definira tenzorskim produktom.

Definicija 5.1 (Tenzorski produkt) Neka postoje dva vektorska prostora V, W dimenzija m, n. Tenzorski produkt prostora V, W se označava $L = V \otimes W$ i to je novi vektorski prostor dimenzije mn. Neka postoji vektor $v \in V, |v\rangle = \sum_i v_i |i\rangle$ i vektor $w \in W, |w\rangle = \sum_j w_j |j\rangle$. Tenzorski produkt vektora $|v\rangle, |w\rangle$ se piše:

$$|v\rangle \otimes |w\rangle = \sum_{i,j} \left(v_i w_j\right) |i\rangle \otimes |j\rangle. \tag{5.5}$$

Tenzorski produkt ima više načina zapisa: $|v\rangle \otimes |w\rangle = |v\rangle|w\rangle = |vw\rangle$.

Stanje sustava s dva qubita opisuje se s:

$$|\psi\rangle = \alpha|00\rangle + \beta|01\rangle + \gamma|10\rangle + \delta|11\rangle. \tag{5.6}$$

Prema tome, u dva qubita se istovremeno mogu pohraniti četiri vrijednosti 0,1,2 i 3. Klasičan sustav s dva bita može istovremeno pohraniti samo jednu vrijednost od 4 moguće. Ovo se još naziva *kvantni paralelizam* i temelj je rada kvantnih algoritama. Kolika je moć kvantnog paralelizma ilustrirano je u tablici 5.1.

Tablica 5.1 Pohranjivanje informacije u više qubita

broj qubita	istovremeno pohranjuje	ukupno
1	(0 i 1)	2
2	(0 i 1)(0 i 1)	4
3	(0 i 1)(0 i 1)(0 i 1)	8
300	(0 i 1)…(0 i 1)	$2^{300} \approx 2 \cdot 10^{90}$

U sustavu s dva qubita iz relacije (5.6) vjerojatnost da se mjerenjem prvog qubita dobije 0 je $|\alpha|^2 + |\beta|^2$. Vjerojatnost da se dobije vrijednost 1 je $|\gamma|^2 + |\delta|^2$. Ako se dobije mjerenjem 0, sustav se nalazi u novom stanju:

$$|\psi'\rangle = \frac{\alpha|00\rangle + \beta|01\rangle}{\sqrt{\alpha^2 + \beta^2}}. \tag{5.7}$$

Općenito, proces mjerenja nekog kvantnog stanja se prikazuje djelovanjem hermitskog operatora koji se naziva projektor. Više o tome može se pogledati u [20].

5.2. Kvantna vrata

Slično logičkim sklopovima u klasičnom računarstvu, u kvantnom računarstvu postoje kvantna vrata i sklopovi. Pod kvantnim vratima se smatraju unitarni operatori dok se pod kvantnim sklopovima smatraju nizovi unitarnih operatora. U nastavku su dane matrice najčešće korištenih operatora koji djeluju na jedan ili na dva qubita.

Pauli-X: $$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad |0\rangle \to |1\rangle, |1\rangle \to |0\rangle \tag{5.8}$$

Pauli-Z: $$X = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad |0\rangle \to |0\rangle, |1\rangle \to -|1\rangle \tag{5.9}$$

Hadamard: $$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \quad |0\rangle \to \frac{|0\rangle + |1\rangle}{\sqrt{2}}|0\rangle, \qquad |1\rangle \to \frac{|0\rangle - |1\rangle}{\sqrt{2}} \tag{5.10}$$

CNOT: $$U_{CN} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad \begin{matrix} |00\rangle \to |00\rangle, & |01\rangle \to |01\rangle, \\ |10\rangle \to |11\rangle, & |11\rangle \to |10\rangle \\ |a, b\rangle \to |a, a \oplus b\rangle \end{matrix} \tag{5.11}$$

Navedeni operatori su unitarni što omogućuje reverzibilnost operacije. Pauli-X operator je analogija klasičnom NOT operatoru. Operatorom Hadamard stanje qubita prelazi u superpoziciju stanja. [20][15]

5.2.1. Kopiranje kvantnih stanja

Dok je u klasičnom računarstvu jednostavno napraviti sklop za kopiranje stanja, u kvantnom računarstvu to nije moguće (vidi *no-cloning* teorem u [20]). To svojstvo se koristi u kvantnoj kriptografiji [21].

5.3. Kvantni algoritmi i njihov utjecaj na asimetričnu kriptografiju

5.3.1. Groverov algoritam

Grover je istraživao efikasne načine pretraživanja NP prostora stanja, ali njegov kvantni algoritam nije mogao ići brže od $O(\sqrt{n})$ koraka, gdje je n broj podatka. Takvo ubrzanje pretraživanja prostora stanja je preslabo ako n raste eksponencijalno te je njegov utjecaj na današnje kriptosustave zanemariv. Ubrzanje pretraživanja se jednostavno može kompenzirati povećanjem duljine kriptografskih ključeva.

Problem pretraživanja se sastoji od traženja indeksa x nekog podatka tako da vrijedi $f(x) = 1$ ako je to podatak koji se traži ili $f(x) = 0$ inače. Za tu potrebu koristi se crna kutija, odnosno unitarni operator O čija je unutarnja struktura nebitna. Djelovanje operatora O na dva qubita je definirano s:

$$|x\rangle|y\rangle \xrightarrow{O} |x\rangle|y \oplus f(x)\rangle. \tag{5.12}$$

Ako se pretpostavi da se traži indeks x, tada djelovanje operatora O na $|x\rangle|0\rangle$ daje $|x\rangle|1\rangle$ što se može shvatiti kao invertiranje drugog qubita. Kada se crnoj kutiji drugi qubit postavlja u superpoziciju dobiva se:

$$|x\rangle|\frac{|0\rangle - |1\rangle}{\sqrt{2}}\rangle \xrightarrow{O} (-1)^{f(x)}|x\rangle|\frac{|0\rangle - |1\rangle}{\sqrt{2}}\rangle. \tag{5.13}$$

Dakle, umjesto invertiranja drugog qubita, napravila se promjena predznaka amplitude stanja sustava. Kako se drugi qubit ne koristi, djelovanje crne kutije može se prikazati s:

$$|x\rangle \xrightarrow{O} (-1)^{f(x)}|x\rangle. \tag{5.14}$$

Neka je N prostor stanja, odnosno skup elemenata u prostoru stanja. Groverov algoritam koristi crnu kutiju O, Hadmardov operator H (5.10) i definira novi operator U_0:

$$|0 \dots 0\rangle \xrightarrow{U_0} -|0 \dots 0\rangle \; i \;\; |j\rangle \xrightarrow{U_0} |j\rangle \;\; za \; j \neq 0. \tag{5.15}$$

Groverov algoritam radi na sljedeći način:

1. Postavi sve qubite u stanje $|0\rangle$
2. Primijeni Hadamardov operator H na sve qubite kako bi dobio stanje $|\psi\rangle = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} |j\rangle$.
3. Slijedno primjenjuj operatore O, H, U_0 i ponovo H: $U_G = HU_0HO$ ukupno $(\pi/4)\sqrt{N}$ puta

Sustav je sada s velikom vjerojatnošću u stanju $|w\rangle = |1\rangle$. Odnosno, $U_G^{\sqrt{N}}|0 \dots 0\rangle \approx |w\rangle$. Budući da svaka primjena operatora U_G označava jednu primjenu operatora O, traženo stanje $|w\rangle$ postići će se za $O(\sqrt{N})$ poziva crne kutije.

Algoritam radi i za slučajeve kada postoji više elemenata za koje je $f(x) = 1$. U tom slučaju algoritam treba $(\pi/4)\sqrt{N/M}$ poziva crne kutije da nađe traženi element, gdje je M broj različitih elemenata za koje je $f(x) = 1$. [15][20]

5.3.2. Shorov algoritam

Shorov algoritam se zasniva na rješavanju problema diskretnog logaritma.

Rješavanjem diskretnog logaritma može se faktorizirati broj N na sljedeći način:

> Prvo se odabere neki $x < N$. Zatim se odredi $nzd(x, N)$. Ako je $nzd(x, N) \neq 1$ nađen je jedan od faktora broja N i postupak je gotov.
>
> Ako je $nzd(x, N) = 1$, nađe se najmanji r za koji vrijedi $x^r \equiv 1(mod\ N)$. (riješi se diskretni logaritam)
>
> Ako je r neparan, odabire se novi broj x i postupak se ponavlja dok r ne bude paran.
>
> Jednom kad je nađen paran r relacija $x^r \equiv 1(mod\ N)$ se primjenom razlike kvadrata može zapisati kao $(x^{r/2} - 1)(x^{r/2} + 1) \equiv 0(mod\ N)$. Odnosno $(x^{r/2} - 1)(x^{r/2} + 1) = kN$ za neki $k \neq 0$. Očito je da k mora biti različit od 0 jer su i faktori $(x^{r/2} - 1)$ i $(x^{r/2} + 1)$ različiti od 0. Nadalje $nzd\left((x^{r/2} - 1), N\right)$ i $nzd\left((x^{r/2} + 1), N\right)$ dati će netrivijalne faktore broja N.

Ulazni parametri: broj N

Izlazni parametri: brojevi p i q za koje vrijedi $N = pq$

```
1: r = 1
2: dok je (r mod 2 == 1)
3:    odaberi x<N
4:    nađi r za koji vrijedi x^r ≡ 1(mod N) // riješi diskretni
                                            // logaritam metodom iz [15]
5: p = nzd((x^(r/2) − 1), N)
6: q = nzd((x^(r/2) + 1), N)
7: vrati p i q
```

Slika 5.1 Shorovog algoritam

Temelj kvantnog algoritma za rješavanje problema diskretnog logaritma je periodičnost niza $x^0(mod\ N), x^1(mod\ N), x^2(mod\ N), \ldots$. Taj niz je periodičan s periodom r, odnosno $x^0(mod\ N) = x^r(mod\ N)$ i r je rješenje problema diskretnog logaritma. Algoritam upotrebljava dva n-qubitna registra i radi na slijedeći način:

Odabere se broj n takav da vrijedi $N^2 < 2^n < 2N^2$.

Izvede se uniformna superpozicija: $2^{-n/2} \sum_{k=0}^{2^n-1} |k\rangle|0\rangle$.

Zatim se napravi modularno potenciranje i dobije se stanje:

$$2^{-n/2} \sum_{k=0}^{2^n-1} |k\rangle|x^k\ mod\ N\rangle. \tag{5.16}$$

Na prvi registar stanja iz (5.16) primjeni se kvantna Fourierova transformacija (QFT [15]) i rezultat je

$$2^{-n} \sum_{jk=0}^{2^n-1} e^{2\pi ijk/2^n} |j\rangle|x^k\ mod\ N\rangle. \tag{5.17}$$

Zbog periodičnosti pozitivna interferencija se događa kada je $j(k + lr)$ blizu višekratnika od 2^n. Dakle, mjerenjem prvog registra dobiva se broj j takav da je $jr/2^n$ blizu cijelog broja. Drugim riječima, algoritam otkriva j takav da je $j/2^n = s/r$ za neki cijeli broj s. Kako bi se našao broj r potrebno je naći razlomak s/r koji aproksimira $j/2^n$. Takvi razlomci pronalaze se upotrebom kontinuirane aproksimacije razlomka. Višestruko ponavljanje aproksimacije razlomka ($2 \log N$ puta) dovoljno je za otkrivanje broja r.

Više o algoritmu za rješavanje diskretnog algoritma pogledati u [15].

Primjer:

Neka se traže faktori broja $N = 35$.

Neka je odabran $x = 2$. Diskretnim logaritmom dobiva se da je $r = 12$, odnosno $2^{12}\ mod\ 35 = 1$. r je paran i može se nastaviti dalje.

Računanjem $nzd\left(\left(x^{r/2} - 1\right), N\right)$ i $nzd\left(\left(x^{r/2} + 1\right), N\right)$ dobivaju se faktori broja N:

$$nzd\left((2^6 - 1), N\right) = 7\ i\ nzd\left((2^6 + 1), N\right) = 5$$

Dolaskom kvantnih računala, od asimetričnih kriptosustava prezentiranih u tablici 4.1, jedino McEliece kriptosustav neće biti probijen. No, McEliece kriptosustav koristi jako velike ključeve te stoga nije prikladan za upotrebu. Shorov algoritam je, dakle, označio kraj asimetrične kriptografije današnjice. Potrebni su asimetrični algoritmi koji se temelje na drugim problemima poput NTRU-a.

6. NTRU kriptosustav

NTRU kriptosustav [22] (engl. *N-th degreeTRUncated polynomial ring*) osmislili su 1996.[9] J. Hoffstein, J. Piper i J. H. Silverman. NTRU je vjerojatnosni kriptosustav koji radi s prstenom polinoma. Temelji se na SVP-u te je zbog toga jedan od najboljih kandidata za korištenje u vrijeme kvantnih računala.

Do danas, razvijeno je nekoliko preinaka prvotnog algoritma uz pomoć kojih se povećala efikasnost i sigurnost NTRU kriptosustava. Zbog velike brzine i ne toliko velike duljine ključeva, NTRU kriptosustav bi mogao zamijeniti današnje asimetrične kriptosustave i prije pojave kvantnih računala.

6.1. Opis NTRU algoritma

6.1.1. Parametri NTRU kriptosustava

NTRU kriptosustav upotrebljava prsten polinoma $R = Z[X]/(X^N - 1)$.

Parametri koji određuju NTRU kriptosustav su [23]:

- N duljina polinoma; polinomi u R su stupnja najviše $N - 1$; prost broj
- q veliki modul; cjelobrojna konstanta
- p mali modul; polinom ili cjelobrojna konstanta
- $\mathcal{D}_f$ prostor privatnog ključa
- $\mathcal{D}_g$ prostor javnog ključa
- $\mathcal{D}_r$ prostor polinoma za prikrivanje (engl. *Blinding Polynomial*)
- $\mathcal{D}_m$ prostor polinoma čistog teksta

Svojstva koja navedeni parametri moraju zadovoljavati su:

1. Parametri (N, p, q) su javni,
2. N mora biti prost broj,
3. p i q moraju biti relativno prosti: $nzd(p, q) = 1$

Prostori polinoma $\mathcal{D}_f$, $\mathcal{D}_g$, $\mathcal{D}_r$ i $\mathcal{D}_m$ definirani su parametrima d_f, d_g, d_r i d_m respektivno i sastoje se od polinoma čiji su koeficijenti iz skupa $\{-1,0,1\}$ ili iz skupa $\{0,1\}$, tzv. „mali polinomi“. U tablici 6.1 detaljnije su definirani pojedini prostori polinoma za binarne i ternarne male polinome.

[9] NTRU je objavljen tek 1998. jer bio odbijen od strane odbora konferencije CRYPTO 97'

Tablica 6.1 Definiranje prostora malih polinoma

Oznaka skupa polinoma	Broj koeficijenata jednakih +1	Broj koeficijenata jednakih -1 (samo za ternarne polinome)
$\mathcal{D}_f$	d_f	$d_f - 1$
$\mathcal{D}_g$	d_g	d_g
$\mathcal{D}_r$	d_r	d_r
$\mathcal{D}_m$	$\geq d_m$	$\geq d_m$

Skup $\mathcal{D}_f$ za ternarne polinome je drugačije definiran radi osiguravanja postojanja inverza. Skup $\mathcal{D}_m$ je skup svih ispravnih reprezentacija čistog teksta kako sigurnost kriptosustava ne bi bila kompromitirana.

Parametri NTRU kriptosustava preporučeni u [23] prikazani su u tablici 6.2.

Tablica 6.2 Preporučeni parametri NTRU kriptosustava [23]

Razina sigurnosti	N	p	q	d_f	d_g	d_r
Umjerena	167	3	128	61	20	18
	167	2	127	45	35	18
Standardna	251	3	128	50	24	16
	251	2	127	35	35	22
Visoka	503	3	256	216	72	55
	503	2	257	155	100	65

6.1.2. Generiranje ključeva

Generiranje ključeva počinje odabirom polinoma $f \in \mathcal{D}_f$ i $g \in \mathcal{D}_g$. Zatim se računaju inverzi polinoma f modulo p i q, f_p i f_q:

$$f \circledast f_p \equiv 1(mod\ p)\ \ i\ \ f \circledast f_q \equiv 1(mod\ q). \tag{6.1}$$

Ako za odabrani f ne postoje inverzi, odabire se novi polinom f dok se ne nađe onaj za kojeg postoje traženi inverzi.

Javni ključ je polinom

$$h = pf_q \circledast g(mod\ q). \tag{6.2}$$

Privatni ključ je par polinoma (f, f_p). [23]

6.1.3. Postupak kriptiranja

Kriptiranje se izvodi tako da se čisti tekst prebaci u polinomni oblik m s koeficijentima modulo p, odnosno $m \in \mathcal{D}_m$. Nakon toga se slučajnim odabirom generira polinom $r \in \mathcal{D}_r$. Upotrebljavajući poruku m, polinom r i javni ključ h računa se kriptirana poruka e [23]:

$$e \equiv r \circledast h + m \ (mod\ q) \tag{6.3}$$

6.1.4. Postupak dekriptiranja

Dekriptiranje počinje upotrebom polinoma f (dijela privatnog ključa):

$$a \equiv f \circledast e (mod\ q). \tag{6.4}$$

Koeficijenti polinoma a moraju biti u intervalu $\left[-\frac{q}{2}, \frac{q}{2}\right]$. Nakon ovog koraka radi se redukcija polinoma a modulo p i računa se originalna poruka m [23]:

$$b \equiv a (mod\ p) \tag{6.5}$$

$$m \equiv f_p \circledast b (mod\ p) \tag{6.6}$$

6.1.5. Primjer

Neka su parametri: $N = 11,\ q = 32,\ p = 3$, $df = 4$, $dg = dr = 3$.

Umjesto binarnog zapisa koristiti će se zapis s oktetima. Primjerice, ako je binarni niz jednak $[0111\ 0000\ 0100\ 0111]$, to će se u oktetima zapisati kao $[112\ 71]$.

Generiranje ključeva:

Neka su slučajnim odabirom generirani f i g kako slijedi:

$$f = -1 + X + X^2 - X^4 + X^6 + X^9 - X^{10}$$
$$g = -1 + X^2 + X^3 + X^5 - X^8 - X^{10}$$

U oktetima, postupkom opisanim u poglavlju 10.5.4, će se to zapisati kao:

$$f = [200\ 28\ 164]\ i\ g = [174\ 87\ 164]$$

Računaju se inverzi od f modulo q i modulo p te javni ključ h:

$$f_p = 1 + 2X + 2X^3 + 2X^4 + X^5 + 2X^7 + X^8 + 2X^9$$

$$f_q = 5 + 9X + 6X^2 + 16X^3 + 4X^4 + 15X^5 + 16X^6 + 22X^7 + 20X^8 + 18X^9 + 30X^{10}$$

$$h \equiv pf_p \circledast g(mod\ 32) = 8 + 25X + 22X^2 + 20X^3 + 12X^4 + 24X^5 + 15X^6 +$$
$$19X^7 + 12X^8 + 19X^9 + 16X^{10}(mod\ 32)$$

Javni ključ će se u oktetima zapisati kao: $h = [70\ 109\ 70\ 97\ 243\ 100\ 224\ 6]$

Kriptiranje:

Pretpostavimo da je poruka m jednaka $[001101]$. Nakon pretvorbe u polinomni oblik (vidi poglavlje 10.5.2) dobiva se $m = -1 + X^2 + X^3 - X^4 + X^9$. Slučajni polinom r neka je $r = -1 + X^2 + X^3 + X^4 - X^5 - X^7$.

Tada je kriptirana poruka e jednaka

$$e \equiv (r \circledast h + m)(mod\ 32) = 14 + 11X + 27X^2 + 24X^3 + +14X^4$$
$$+16X^5 + 30X^6 + 7X^7 + 26X^8 + 6X^9 + 18X^{10}(mod\ 32).$$

U oktetima kriptirana poruka jednaka je: $e = [114\ 247\ 135\ 67\ 199\ 209\ 164\ 6]$

Dekriptiranje:

Pri dekripciji poruke e računaju se polinomi a, b i krajnji polinom c koji je dekriptirani izvorni tekst.

$$a \equiv f \circledast e(mod\ 32) = 3 - 7X - 10X^2 - 11X^3 + 10X^4 +$$
$$+7X^5 + 6X^6 + 7X^7 + 5X^8 - 3X^9 - 7X^{10}(mod\ 32).$$

Važno je napomenuti da pri redukciji polinoma modulo 32 koeficijenti budu iz intervala [-15,16], a ne iz intervala [0,31].

$$b \equiv a\ (mod\ 3) = -X - X^2 + X^3 + X^4 + X^5 + X^7 - X^8 - X^{10}(mod\ 3)$$

$$c \equiv f_p \circledast b = -1 + X^3 - X^4 - X^8 + X^9 + X^{10}(mod\ 3) = m$$

6.1.6. Matematička pozadina

Kada Alice kriptira poruku m Bobovim javnim ključem h, ona generira polinom r koji Bob ne zna. No, Bob izračunavanjem $a = f \circledast e (mod\ q)$ zapravo radi sljedeće:

$$a \equiv f \circledast e (mod\ q)$$

$$\equiv f \circledast (r \circledast h + m) (mod\ q) \quad | jer\ e \equiv r \circledast h + m\ (mod\ q)$$

$$\equiv f \circledast \left(r \circledast pf_q \circledast g + m\right) (mod\ q) \quad | jer\ h \equiv pf_q \circledast g\ (mod\ q)$$

$$\equiv pr \circledast g + f \circledast m\ (mod\ q) \quad | jer\ f \cdot f_q \equiv 1 (mod\ q)$$

Budući da su koeficijenti polinoma r, g, f i m mali, onda su i koeficijenti produkata $r \circledast g$ i $f \circledast m$ također mali, barem u odnosu na q. Iz toga slijedi, ako su parametri dobro odabrani, da koeficijenti polinoma $pr \circledast g + f \circledast m$ već leže unutar intervala $\left[-\frac{q}{2}, \frac{q}{2}\right]$, pa redukcija modulo q nema nikakvog utjecaja. Drugim riječima, pri izračunu polinoma a, polinom a je zapravo identičan polinomu $pr \circledast g + f \circledast m$. Nadalje, kada se radi rekucija polinoma a modulo p, zapravo se radi redukcija polinoma $pr \circledast g + f \circledast m$ modulo p i time se dobiva $b \equiv f \circledast m (mod\ p)$. Sada je poznat polinom b, polinom f i njegov inverz modulo p (f_p) te je konačni korak množenje polinoma b i f_p budući da je $f \circledast f_p \equiv 1 (mod\ p)$. Dakle, $c = f_p \circledast b = f_p \circledast f \circledast m \equiv m\ (mod\ p)$. [23]

6.1.7. Pogrešno dekriptiranje

U prethodnom poglavlju opisano je što se radi u svakom koraku dekriptiranja te je rečeno da koeficijenti polinoma $f \circledast e = pr \circledast g + f \circledast m$ u većini slučajeva leže unutar intervala $\left[-\frac{q}{2}, \frac{q}{2}\right]$ te se s tom pretpostavkom razmatraju daljnji koraci dekriptiranja. U slučajevima kada koeficijenti polinoma $pr \circledast g + f \circledast m$ uistinu jesu unutar navedenog intervala, dekriptiranje će biti uspješno. Onda kada to nije slučaj, rezultat dekriptiranja biti će pogrešan.

Pogrešno dekriptiranje je najveća mana NTRU kriptosustava i do pojave NAEP-a (vidi poglavlje 8) pogrešno dekriptiranje se upotrebljavalo za napad [25] na privatni ključ.

Neka je $\max a = \max_i a_i$, $\min a = \min_i a_i$ i neka je širina polinoma a jednaka $width(a) = \max a - \min a$.

Pogreška pri dekriptiranju će se dogoditi ako je $width(pr \circledast g + f \circledast m) \geq q$ što se označava kao *gap failure*, ili ako $width(pr \circledast g + f \circledast m) < q$, a polinom je reduciran u krivi interval što se označava kao *wrap failure*. U idućem poglavlju

opisan je postupak za određivanje točnog intervala kojim se sprječava *wrap failure*, centriranje polinoma. No, još uvijek ne postoje metode za sprečavanje *gap failura*.

Vjerojatnost pogrešnog dekriptiranja (*gap failure*) jednaka je [23]

$$P_{err} = 2^{-a}, uz$$
$$a = N \cdot erfc\left(\frac{q-2}{8p\sqrt{d}}\sqrt{3}\right). \tag{6.7}$$

6.1.8. Centriranje polinoma

Ako se u NTRU kriptosustavu upotrebljavaju binarni polinomi umjesto ternarnih, gotovo je potpuno sigurno da će doći do pogreške u dekriptiranju. Kako bi se spriječio *wrap failure*, nakon prvog koraka dekriptiranja, polinom a se mora reducirati na interval $[A, A+q-1]$.

Vrijednost A računa se na sljedeći način:

$$A = \frac{p(1) \cdot r(1) \cdot g(1) + f(1) \cdot I}{N} - \frac{q}{2}$$
$$I = e(1) - r(1) \cdot h(1)\ (mod\ q) \tag{6.8}$$

Za NTRU kod kojeg se upotrebljavaju ternarni polinomi $(p = 3)$, vrijednost A jednaka je $-\frac{q}{2}$, odnosno nije potrebno centrirati polinom. [23]

6.2. Varijante NTRU kriptosustava

Varijante NTRU kriptosustava predstavljaju promjene pojedinih dijelova kriptosustava:

- oblik polinoma f
- oblik malog modula p
- oblik polinoma m

Navedene promjene se mogu koristiti zasebno ili se mogu međusobno kombinirati.

6.2.1. NTRU s $p = 2$

Ova inačica NTRU-a koristi binarne polinome budući da se polinomi reduciraju modulo 2. Ovaj način je intuitivan i ne zahtijeva promjenu iz binarne reprezentacije u ternarnu. No, NTRU s ovakvim parametrima je sporiji prilikom generiranja ključeva i dekriptiranja. Generiranje ključeva je sporije jer u ovom slučaju parametar q mora biti prost broj te je inverz modulo q puno zahtjevnija operacija nego dosad. Osim toga, prilikom dekriptiranja potrebno je centrirati polinom a kako bi dekriptiranje bilo uspješno. [23]

6.2.2. NTRU s $f = 1 + pF$

U ovoj varijanti NTRU kriptosustava za privatni ključ f vrijedi $f = 1 + pF$, gdje je $F \in \mathcal{D}_f$. Na ovaj način osiguralo se da inverz polinoma f modulo p uvijek postoji te je on jednak: $f_p = 1(mod\ p)$.

Računanje inverza polinoma f modulo p se više ne mora izvoditi, a isto vrijedi i za zadnji korak prilikom dekriptiranja: $f_p \circledast b(mod\ p) = b(mod\ p) = m$.

Ovim načinom skratilo se vrijeme potrebno za generiranje ključeva i dekriptiranje te duljina privatnog ključa budući da polinom f_p više nije potrebno pamtiti.

Za parametar p može se uzeti 2 ili 3 te se ovisno o tome upotrebljavaju binarni ili ternarni polinomi. [23]

6.2.3. NTRU s $p = 2 + X$

U ovoj varijanti NTRU kriptosustava modul p više nije konstanta već polinom. Za parametar q se uzima potencija broja 2 kao i u NTRU-u s $p = 3$, dakle $q = 2^r$ kako bi brzina kriptosustava bila što veća. Može se koristiti u kombinaciji s prethodnom inačicom s $f = 1 + pF$ oblikom privatnog ključa ili u standardnom NTRU-u. Ovdje dodatni vremenski trošak predstavlja reduciranje polinoma modulo p. [23]

7. Sigurnost NTRU kriptosustava

7.1. NTRU rešetka [26]

Modularna rešetka (engl. *Modular Lattice* - ML) s parametrima $n = 2N$ i q je cjelobrojna rešetka dimenzije n generirana retcima bilo koje $n \times n$ matrice oblika

$$L = \begin{bmatrix} b & \cdots & 0 & h_{11} & \cdots & h_{1N} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & b & h_{N1} & \cdots & h_{NN} \\ 0 & \cdots & 0 & q & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & \cdots & q \end{bmatrix} \tag{7.1}$$

uz $|h_{ij}| \leq q/2$. Pišemo $L_{ML} = rowspan(L)$.

Često se modularne matrice zapisuju kao

$$L = \begin{bmatrix} bI & h \\ 0 & qI \end{bmatrix}, \tag{7.2}$$

gdje je I $n \times n$ jedinična matrica, 0 $n \times n$ nul-matrica, a h $n \times n$ cjelobrojna matrica. Koeficijent b naziva se konstanta balansa (engl. *balancing constant*).

Neka su $f(X)$ i $g(X)$ dva polinoma i neka je $[f, g]$ vektor dimenzije $2N$ formiran od koeficijenata tih polinoma: $[f_0, f_1, \dots, f_{N-1}, g_0, g_1, \dots, g_{N-1}]$. Neka je $M(X) \in Z_q[X]$ polinom stupnja N. Svaki polinom $h(X)$ u kvocijentnom prstenu $Z_q[X]/M(X)$ može se upotrijebiti za formiranje modularne rešetke L_h:

$$L_h = \{[F, G]: F(X)h(X) = G(X)\ u\ Z_q[X]/M(X)\}. \tag{7.3}$$

i-ti redak h matrice sastoji se od koeficijenata ostatka dijeljenja $X^i h(X)$ sa $M(X)$. Za NTRU važan slučaj je za $M(X) = X^N - 1$. Tada je h konvolucijska (cirkularna) matrica sastavljena od koeficijenata polinoma $h(X)$.

Za slučaj kada je matrica h iz (7.2) oblika:

$$h = \begin{bmatrix} h_0 & h_1 & \dots & h_{N-1} \\ h_{N-1} & h_0 & \dots & h_{N-2} \\ \vdots & \ddots & \ddots & \vdots \\ h_1 & \dots & h_{N-1} & h_0 \end{bmatrix}, uz\ |h_i| \leq q/2 \tag{7.4}$$

rešetka L_{ML} se naziva konvolucijska (cirkularna) modularna rešetka (engl. *Convolution Modular Lattice* – CML).

Cikličko svojstvo konvolucijske modularne rešetke označava da za svaki $v = [a_0, a_1, \dots, a_{N-1}, b_0, b_1, \dots, b_{N-1}] \in L_{CML}$, svi vektori dobiveni cikličkim pomakom koordinata od v su također sadržani u L_{CML}.

Determinanta konvolucijske modularne rešetke je zbog cikličkog svojstva uvijek jednaka

$$det(L) = (bq)^N, \tag{7.5}$$

gdje je b konstanta balansa.

Ako konvolucijska modularna rešetka

$$L_{CML} = rowspan\left(L = \begin{bmatrix} bI & h \\ 0 & qI \end{bmatrix}\right) \tag{7.6}$$

sadrži vektor oblika

$$[f, g] = [f_0, f_1, \dots, f_{N-1}, g_0, g_1, \dots, g_{N-1}], \tag{7.7}$$

tada se rešetka iz relacije (7.6) naziva NTRU rešetka, oznake L_{NTRU}.

7.2. Napad rešetkama

7.2.1. Balansiranje CVP-a u modularnim rešetkama

Neka je (L, a) CVP u modularnoj rešetki L i neka je $v \in L$ rješenje.

Vektor a napiše se u obliku $[a_1, a_2]$ tako da a_1 i a_2 svaki imaju po N koordinata. Jednako tome, vektor v napiše se u obliku $[v_1, v_2]$.

Ako se konstanta balansa b u matrici od L zamijeni s b_{new} kako bi se formirala nova modularna rešetka L_{new}, tada CVP (L_{new}, a_{new}) ima rješenje v_{new}, gdje je $a_{new} = \left[\frac{b_{new}}{b} a_1, a_2\right]$ i $v_{new} = \left[\frac{b_{new}}{b} v_1, v_2\right]$. Za dobro odabrani b vektor v_{new} će biti jako blizu vektoru a_{new} i Gaussova heuristika se može upotrijebiti kako bi se potvrdilo da je to vjerojatno najbliži vektor.

U praksi je najlakše riješiti CVP ako su polovice matrice od L balansirane. Za CVP modularne rešetke se kaže da je balansiran ako rješenje $v = [v_1, v_2] \in L$ zadovoljava $\|v_1 - a_1\| \approx \|v_2 - a_2\|$.

Često je moguće upotrijebiti neko znanje o obliku vektora v kako bi se lakše odredila konstanta balansa. Stoga se pri analizi sigurnosti kao pretpostavku mora uzeti da napadač zna kako balansirati CVP. [27]

7.2.2. Osnovni CVP omjeri

Ako rešetka L ima bazu koja se sastoji od n jednako dugih ortogonalnih vektora, tada svaki od tih n vektora baze ima duljinu jednaku $det(L)^{1/dim(L)}$. S rešetkama koje imaju takvu bazu je posebno lako raditi. Za CVP (L, a) u kojem je traženi vektor bliže vektoru a nego prema Gaussovoj heuristici vrijednost ρ opisuje koliko je teško doći do rješenja:

$$\rho = \rho(L, a) = \lambda(L, a)/det(L)^{1/dim(L)}. \tag{7.8}$$

Ako je $v \in L$ rješenje CVP-a, što je manja vrijednost od ρ to je vektor v bliži vektoru a nego ostalim vektorima u rešetci, odnosno lakše je riješiti CVP.

U praksi se pokazalo da je bolja konstanta koja pokazuje ovisnost vremena rješavanja o dimenziji rešetke konstanta c:

$$c = \sqrt{4\pi e\|f\|\|g\|/q}. \tag{7.9}$$

Neka je L modularna rešetka dimenzije $n = 2N$ i modula q. Druga vrijednost koja utječe na teškoću rješavanja CVP-a je omjer:

$$a = \frac{N}{q}. \tag{7.10}$$

U praksi se pokazalo da održavanje konstante a i istovremeno povećanje konstante c drastično povećava vrijeme potrebno za redukciju rešetke. Isto tako pokazalo se da održavanje konstante c i povećanje konstante a neznatno smanjuje vrijeme potrebno za redukciju rešetke. [27]

7.2.3. Napad rešetkom na NTRU privatni ključ

CVP konvolucijske modularne rešetke formirane od koeficijenata $h(X)/p\ (mod\ q)$ ima traženi vektor $v = [bv_1, v_2]$ formiran od koeficijenata $[f(X), g(X)]$. Za slučaj kada se koristi $f = 1 + pF$ traženi vektor ima oblik $[F(X), g(X)]$.

7.2.4. Napad rešetkom na poruku

CVP se može formirati i preko javnog ključa gdje je traženi vektor $v = [bv_1, v_2]$ formiran od koeficijenata $[r(X), m(X)]$. To je napad na izvornu poruku m, koji je nešto lakši od napada na ključ.

7.2.5. Heurističko vrijeme rješavanja CVP-a u modularnim rešetkama

Neka je L modularna rešetka dimenzije $n = 2N$, modula q i neka je (L, v) balansirani CVP rešetke L. Prosječno vrijeme T potrebno za rješavanje (L, v) je eksponencijalno u dimenziji s konstantama ovisnima o vrijednostima c i a opisanima ranije:

$$\log T \approx AN + B, \tag{7.11}$$

gdje je $\alpha = \alpha(c, a)$ i $\beta = \beta(c, a)$, $c = c(L, v)$, $a = a(L)$. Uoči da $\log T$ zapravo predstavlja bitovnu sigurnost (engl. *bit security*).

U tablici 7.1 prikazani su eksperimentalni rezultati iz [27] za sigurnost NTRU-a s obzirom na napade rešetkama.

Tablica 7.1 Sigurnost NTRU-a s obzirom na napade rešetkama

c	a	A	B	sigurnost za $N = 251$
1.73	0.53	0.3563	−2.263	87 bitova
2.6	0.8	0.4245	−3.440	103 bitova
3.7	2.7	0.4512	+0.218	113 bitova
5.3	1.4	0.6492	−5.436	158 bitova

7.3. Napad čovjek u sredini [28]

Napad čovjek u sredini (engl. *Meet In The Middle attack* – MITM) kao i u drugim kriptosustavima, smanjuje prostor pretraživanja s drugim korijenom.

Ideja MITM napada je naći polinom f u obliku $f = f_1 + f_2$, tako da polinom f_1 ima prvih $N/2$ koeficijenata jednakih korespodentnim koeficijentima u polinomu f, a polinom f_2 ima zadnjih $N/2$ koeficijenata jednakih korespodentnim koeficijentima u polinomu f. Dakle polinomi f_1 i f_2 su zapravo jednaki polinomu f samo što je kod f_1 zadnjih $N/2$ koeficijenata promijenjeno u 0, a kod f_2 prvih $N/2$ koeficijenata je promijenjeno u 0. Oba polinoma f_1 i f_2 imaju $d_f/2$ jedinica. Iako f ne mora imati $d_f/2$ koeficijenata jednakih 1 na prvoj polovici polinoma, poznato je da je dovoljna jedna rotacija da se postigne takav raspored koeficijenata. Upotrebljavajući relaciju $f \circledast h \equiv g \ (mod\ q)$ dobiva se

$$(f_1 + f_2) \circledast h \equiv g \ (mod\ q)$$
$$f_1 \circledast h \equiv g - f_2 \circledast h (mod\ q) \tag{7.12}$$

Budući da polinom g ima koeficijente iz $\{0,1\}$ ili $\{-1,0,1\}$, koeficijenti polinoma $f_1 \circledast h$ i $-f_2 \circledast h$ mogu se samo razlikovati za 0, 1 ili 2. S time u vidu napadač traži par (f_1, f_2) takav da isti koeficijenti imaju približno istu vrijednost.

7.3.1. Postupak napada

Napadač odabire cijeli broj k takav da je

$$2^k > \binom{N/2}{d_f/2}. \tag{7.13}$$

Prvi korak je obrada $\binom{N/2}{d_f/2}$ različitih polinoma f_1. Svaki se f_1 stavlja u tablicu prema najznačajnijem bitu u prvih k koeficijenata izraza $f_1 \circledast h(mod\ q)$. Svako polje u tablici je referencirano sa $\{0,1\}^k$ i postoji 2^k „odjeljaka“ od kojih će oko $\binom{N/2}{d_f/2}$ biti zauzeto.

Na primjer, ako je $k = 4$, $q = 32$ i prva 4 koeficijenta od $f_1 \circledast h(mod\ q)$ su: 25 (11001), 7 (00111), 12 (01100) i 30 (11110) onda polinom f_1 pripada poziciji (1,0,0,1).

Drugi korak je obrada svih mogućih polinoma f_2. Za svaki f_2 se provjeri da li korespondira zauzetoj poziciji u tablici.

Ako su f_1 i f_2 točni vrijedit će $(f_1 \circledast h)_i = \{-1,0,1\} - (f_2 \circledast h)_i \ (mod\ q), \forall i$. Zbog toga se ne provjeravaju samo pozicije dobivene od najznačajnijih bitova koeficijenata od $-f_2 \circledast h(mod\ q)$, nego se provjerava i za pozicije kada se koeficijentima doda 1 (odnosno doda ili oduzme 1 ako je riječ o ternarnim polinomima).

Na primjer, ako je s prethodnim parametrima uz $p = 2$ (binarni polinomi) prvih 4 koeficijenata od $-f_2 \circledast h(mod\ q)$ jednako 11 (01011), 23 (10111), 14 (01110) i 2 (00010) provjerit će se samo pozicija (0,1,0,0). No, ako je prvih 4 koeficijenata jednako 15 (01111), 31 (11111), 17 (10001) i 10 (01010) provjerit će se pozicije (0,1,1,0), (1,1,1,0), (0,0,1,0) i (1,0,1,0).

Kada f_2 pogodi zauzetu poziciju izračuna se $f = f_1 + f_2$, provjeri se je li $f \circledast h\ (mod\ q)$ binarni (odnosno ternarni) polinom. Ako je, postupak je gotov, inače se nastavlja s novim f_2.

7.3.2. Vremenska i prostorna složenost

Neka je T_c vrijeme potrebno za konvolucijsko množenje $f_1 \circledast h(mod\ q)$. Vrijeme potrebno za izračun $f \circledast h(mod\ q)$ neće biti dulje od $2T_c$. Neka je T_s vrijeme potrebno za pretraživanje tablice, ili pisanje i čitanje u tablici.

Vrijeme potrebno za prvi korak napada je najviše:

$$T_1 = \binom{N/2}{d_f/2}(T_c + T_s). \tag{7.14}$$

Očekivano vrijeme trajanja drugog dijela napada će biti najviše:

$$\begin{aligned} T_2 = \#f_2 \cdot [T_c &+ (očekivani\_br\_pozicija\_po\_f_2) \cdot T_s \\ &+ (očekivani\_br\_pogodaka\_po\_f_2) \cdot T_c] = \\ &= \binom{N/2}{d_f/2}\left(T_c + \frac{2k}{q}T_s + \frac{\binom{N/2}{d_f/2}}{2^k}T_c\right). \end{aligned} \tag{7.15}$$

Ako je μ_f memorija potrebna za spremanje polinoma f_1 i ostalih informacija, a μ_0 dodatni trošak potreban zbog infrastrukture spremišta podataka ukupna memorija potrebna za MITM napad je približno jednaka:

$$\mu \approx \binom{N/2}{d_f/2}\mu_f + \mu_0. \tag{7.16}$$

μ_0 se povećava s k, ali ne eksponencijalno i μ_f se povećava s k, ali ne brže od k.

Ako se u drugom koraku umjesto $f \circledast h\ (mod\ q)$ izvodi $(f_1 + f_2) \circledast h\ (mod\ q)$ vremenska složenost se smanjuje na približno $\frac{\binom{N/2}{d_f/2}}{\sqrt{N}}$.

7.4. Hibridni napad

Hibridni napad prezentiran na CRYPTO '07 [30] kombinacija je MITM napada i napada rešetkom.

CVP se može efikasno riješiti kada je zadana točka jako blizu nekom vektoru rešetke. CVP se rješava u vremenu t uz pomoć skupa S koji ima svojstvo da sadržava barem jednu točku $v_0 \in S$ koja je jako blizu nekom vektoru rešetke. Tada se v_0 može naći u vremenu $O(|S|t)$ pretraživanjem prostora. Ako se skup S

prezentira kao $S = S' \oplus S'$ (za svaki $(v, v') \in S \cdot S'$ postoji $v'' \in S'$ takav da je $v = v' + v''$), tada postoji efikasan MITM algoritam koji nalazi točku v_0 u vremenu $O\left(t \cdot \sqrt{|S|}\right)$. [29]

7.5. Određivanje parametara NTRU kriptosustava

U [31] predložen je algoritam za generiranje parametara NTRU kriptosustava za zadanu razinu sigurnosti. Algoritam na ulaz prima parametar sigurnosti k. Navedeni algoritam za binarne polinome (p=2) je prikazan na slici 7.1. Na sličan način radi se i generiranje parametara za ternarne polinome (p=3). [29]

Ulazni parametri: razina sigurnosti k

Izlazni parametri: (N, q, df, dg, dr, dm)

1: odaberi prvi prost broj N za koji vrijedi N>3k+8
2: odaberi d kao najmanji integer koji zadovoljava $\frac{1}{\sqrt{N}}\binom{N/2}{d/2} > 2^k$
3: postavi df = dr = d, dg = N/2
4: odaberi dm kao najveći integer koji zadovoljava $2^{N-1}\sum_{i=0}^{dm}\binom{N}{i} < 2^{-40}$
5: **ako** $\frac{1}{\sqrt{N}}\binom{N/2}{dm} < 2^k$
6: postavi N na prvi sljedeći prost broj i vrati se na korak 2
7: postavi q na prvi prost broj koji zadovoljava $q > 4d + 1$
8: **dok** (red od q modulo N)!=N-1 && (red od q modulo N)!=(N-1)/2
9: povećaj q na sljedeći prost broj
10: iz tablice 6.3 odredi parametre A i B
11: provjeri vrijedi li nejednakost

$$AN - B - \max\left[\log_2\left(1 - \left(1 - \prod_{i=0}^{d-1}\left(1 - \frac{r}{\sqrt{N-i}}\right)\right)^N\right) + \frac{Ar}{2}\right] < k$$

12: **ako** je uvjet zadovoljen vrati (N, q, df, dg, dr, dm)
13: **inače** povećaj N na sljedeći prost broj i vrati se na korak 2

Slika 7.1 Algoritam za generiranje NTRU parametara za binarne polinome

7.5.1. Parametri preporučeni u P1363 standardu

U tablicama 7.2, 7.3 i 7.4 prikazani su preporučeni parametri iz [27] s obzirom na razinu sigurnosti i uvjet optimalnosti.

Tablica 7.2 Prostorno optimirani NTRU parametri

Razina sigurnosti	Naziv parametara	N	p	q	$d_f = d_r$	d_g
112	ees401ep1	401	3	2048	113	133
128	ees449ep1	449	3	2048	134	149
192	ees653ep1	653	3	2048	194	217
256	ees853ep1	853	3	2048	268	284

Tablica 7.3 Vremenski i prostorno optimirani NTRU parametri

Razina sigurnosti	Naziv parametara	N	p	q	$d_f = d_r$	d_g
112	ees541ep1	541	3	2048	49	180
128	ees613ep1	613	3	2048	55	204
192	ees887ep1	887	3	2048	81	295
256	ees1171ep1	1171	3	2048	106	390

Tablica 7.4 Vremenski optimirani NTRU parametri

Razina sigurnosti	Naziv parametara	N	p	q	$d_f = d_r$	d_g
112	ees659ep1	659	3	2048	38	219
128	ees761ep1	761	3	2048	42	253
192	ees1087ep1	1087	3	2048	63	362
256	ees1499ep1	1499	3	2048	79	499

8. NAEP/SVES-3 shema kriptiranja

Kao i RSA, NTRU kriptosustav sam po sebi nije potpuno siguran protiv napada odabranim kriptiranim tekstom (engl. *Chosen Ciphertext Attack*) te se zbog toga uvode sheme kriptiranja.

Shema kriptiranja koja sprječava napad odabranim čistim tekstom se označava s IND-CPA (engl. *INDistinguishability under Chosen Plaintext Attack*). Shema kriptiranja koje sprječava napad odabranim kriptiranim tekstom označava se kraticom IND-CCA (engl. *INDistinguishability under Chosen Ciphertext Attack*). U praksi je IND-CCA teško ostvariti.

Za NTRU su se sve predložene sheme kriptiranja do objave NAEP-a (*NTRU Assymetric Encryption Padding* [31]) 2004. g. pokazale nesigurnima. Nakon analiziranja NAEP-a jedna je instanca NAEP sheme (SVES-3 – *Shortest Vector Encryption Sheme*, third revision) u prosincu 2008. godine prihvaćena kao standard za kriptografske metode temeljene na rešetkama: ANSI/IEEE P1363.1-2008. (*Standard Specification for Public-Key Cryptographic Techiques Based on Hard Problems over Lattices*). [27]

8.1. Opis NAEP/SVES-3 sheme [27]

SVES-3 shema upotrebljava dvije *hash* funkcije, G i H. Hash funkcija G se naziva *Blinding Polynomial Generation Method* – BPGM (generiranje maskirajućeg polinoma). BPGM upotrebljava IGF (*Indeks Generation Function*) metodu koja nekoliko puta za redom upotrebljava hash funkciju (npr. SHA-256). Ulazni string $(ID\|m\|b)$ se proširuje na string $(ID\|m\|b\|hTrunc)$, gdje je $hTrunc$ nekoliko bitova javnog ključa h. Funkcija H se naziva *Mask Generation Function* (MGF) i upotrebljava hash funkciju (npr. SHA-256).

Parametri SVES-3 sheme su maksimalna duljina poruke u bitovima (`maxLen`) i duljina slučajnog stringa u bitovima (`bLen`). Duljina cijele poruke (u bitovima) se računa kao:

$$nLen = bLen + (\log_2(maxLen) + 1) + maxLen. \tag{8.1}$$

8.1.1. Kriptiranje

Prije kriptiranja poruke M, računa se duljina poruke M u bitovima (`MLen`) i odabire se slučajni `string` (niz znakova) `b` duljine `bLen` u bitovima. Generira se polinom r (*blinding polynomial*):

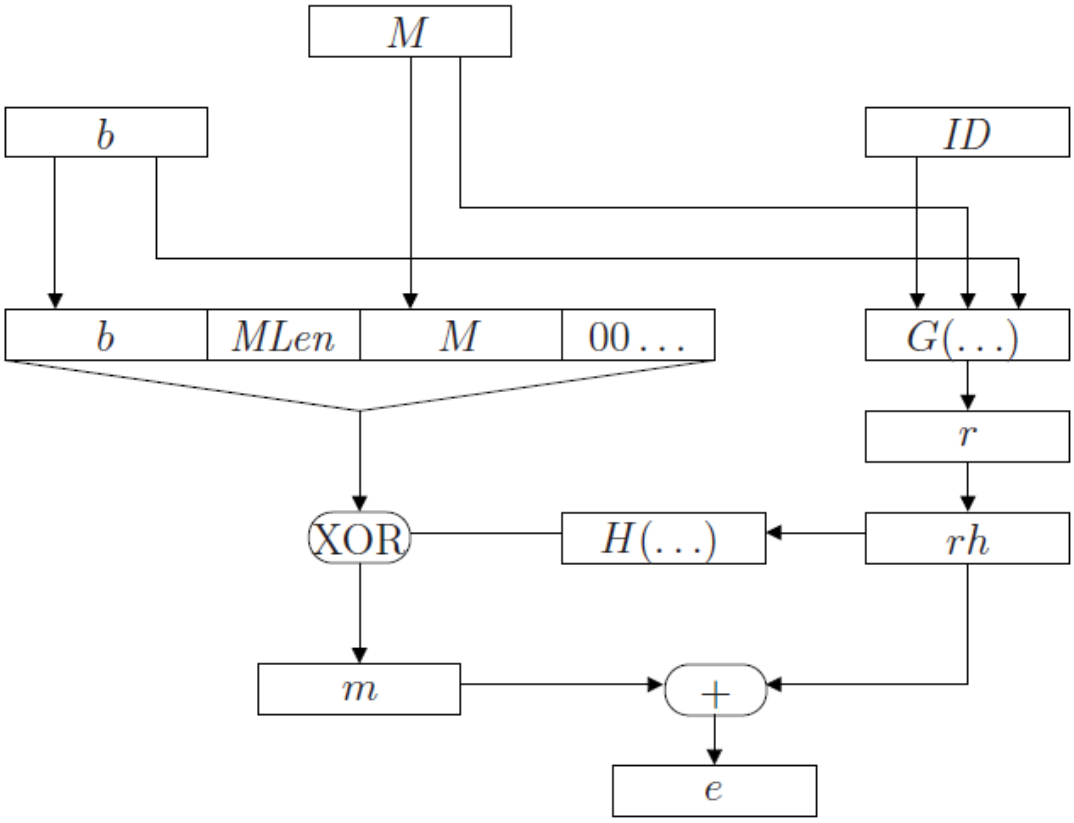
$$r = G(ID \| M \| b), \tag{8.2}$$

gdje je ID je broj koji identificira upotrijebljene parametre.

Poruka M se popuni kao $M = b \| MLen \| M \| 00 \ldots$ kako bi se dobila poruka duljine $nLen$. Napravi se logička ekskluzivno-ili operacija nad M i $H(rh)$:

$$m = M \oplus H(rh). \tag{8.3}$$

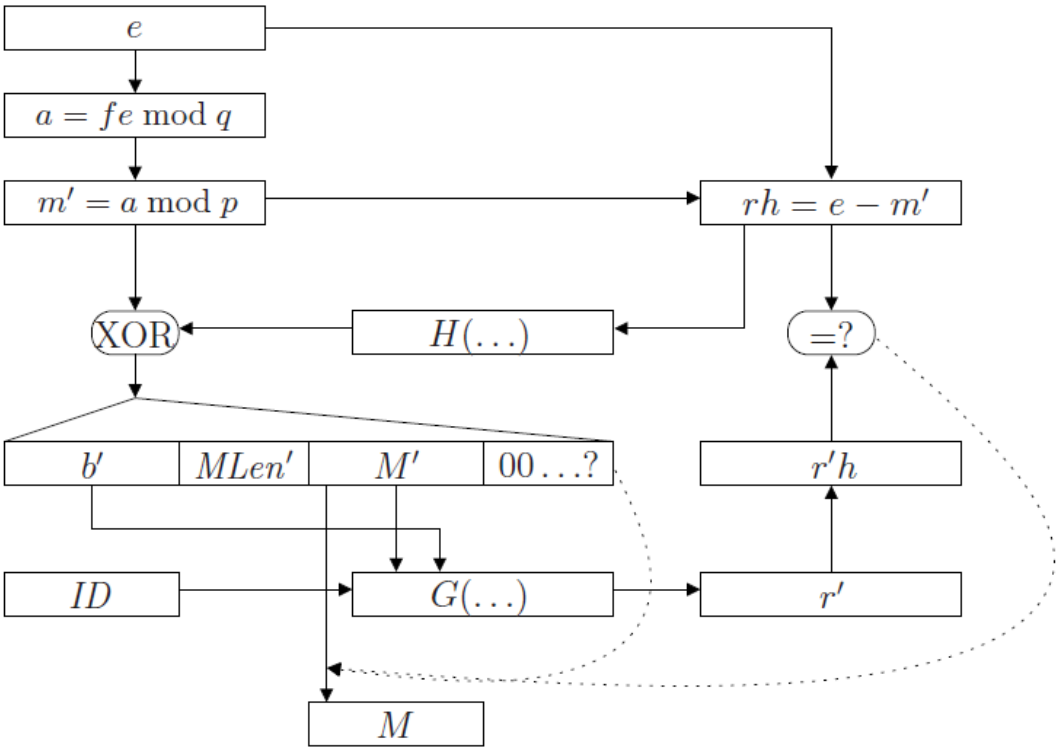
Dobivena poruka m se kriptira standardnim NTRU postupkom kriptiranja.



Slika 8.1 Shema kriptiranja u NAEP/SVES-3

8.1.2. Dekriptiranje

Dekriptira se kriptirani tekst e u polinom m'. Izračuna se razlika $rh = e - m'$ i ekskluzivno-ili nad m' i $H(rh)$ kako bi se dobio string duljine $nLen$. Dobiveni string se interpretira kao $(b'\|MLen'\|M'\|trunc)$. Provjeri se, sastoji li se $trunc$ samo od nula i je li $MLen'$ duljina poruke M'. Izračuna se $r' = G(ID\|M'\|b')$ i provjeri se je li $r'h$ jednako rh. Ako su sve provjere prošle vrati M' kao dekriptiranu poruku.



Slika 8.2 Shema dekriptiranja u NAEP/SVES-3

9. NTRUSign

Nakon probijenog NSS-a (NTRU Signature Sheme [33][34]) tvrtka NTRU Cryptosystems predložila je NTRUSign [35]. NtruSign se temelji na GGH (Goldreich, Goldwasser, Halevi) digitalnom potpisu uz upotrebu NTRU-rešetke.

9.1. GGH digitalni potpis

GGH radi s rešetkama L u Z^n. Privatni ključ je nesingularna matrica $n \times n$ u Z, $R \in M_n(Z)$ sa kratkim vektorima retcima. R je odabran kao perturbacija umnoška jediničnih matrica tako da su njeni vektori gotovo ortogonalni. Točnije $R = kI_n + E$, uz $k = 4\lfloor\sqrt{n} + 1\rfloor + 1$, gdje zagrada $\lceil x \rfloor$ predstavlja cijeli broj najbliži broju x i svaki element $n \times n$ matrice E je odabran uniformno iz skupa $\{-4, \ldots, 3\}$. Rešetka L je razapeta vektorima retcima matrice R. Osoba koja potpisuje može dobro aproksimirati CVP u rešetki L poznavajući matricu R. Baza R je privatni ključ, dok je R transformirana u nereduciranu bazu B javni ključ.

Neka je vektor $m \in Z^n$ sažetak poruke koju se želi potpisati. Potpisivač, primjenjuje Babaijevu tehniku zaokruživanja za aproksimaciju CVP problema kako bi dobio vektor rešetke blizu vektora m:

$$s = \lceil mR^{-1} \rfloor R \tag{9.1}$$

Za provjeru potpisa sažetka m, prvo treba provjeriti je li $s \in L$ upotrebom javne baze rešetke (B) i zatim provjeriti je li udaljenost $\|s - m\|$ dovoljno mala.

9.2. NTRUSign

NTRUSign je specifična instanca GGH digitalnog potpisa sa rešetkama iz NTRU sheme kriptiranja.

Parametri su slični kao i kod NTRU kriptiranja: $N, q, R = Z[X]/(X^N - 1)$. Izračuna se četvorka $(f, g, F, G) \in R^4$ tako da vrijedi $f \circledast G - g \circledast F = q$ i f je invertibilan modulo q. Polinomi f i g su mali polinomi s binarnim koeficijentima sa definiranim brojem jedinica, jednako kao u NTRU kriptosustavu. Polinomi F i G imaju veće koeficijente od polinoma f i g, ali još uvijek manje od broja q. (f, g, F, G) je privatni ključ. Privatni ključ (privatna baza rešetke) je slijedeća $2N \times 2N$ matrica:

$$R = \begin{bmatrix} f_0 & f_1 & \cdots & f_{N-1} & g_0 & g_1 & \cdots & g_{N-1} \\ f_{N-1} & f_0 & \cdots & f_{N-2} & g_{N-1} & g_0 & \cdots & g_{N-2} \\ \vdots & \ddots & \ddots & \vdots & \vdots & \ddots & \ddots & \vdots \\ f_1 & \cdots & f_{N-1} & f_0 & g_1 & \cdots & g_{N-1} & g_0 \\ F_0 & F & \cdots & F_{N-1} & G_0 & G_1 & \cdots & G_{N-1} \\ F_{N-1} & F_0 & \cdots & F_{N-2} & G_{N-1} & G_0 & \cdots & G_{N-2} \\ \vdots & \ddots & \ddots & \vdots & \vdots & \ddots & \ddots & \vdots \\ F_1 & \cdots & F_{N-1} & F_0 & G_1 & \cdots & G_{N-1} & G_0 \end{bmatrix} \quad (9.2)$$

Jedan redak matrice R dovoljan je da se otkrije cijeli privatni ključ. Javni ključ je $h = f_q \circledast g(mod\ q)$, gdje je f_q inverz polinoma f modulo q. Važno svojstvo javnog ključa je $f \circledast h \equiv g\ (mod\ q)$.

Sažetak poruke treba biti oblika $m \in \{0, \dots, q-1\}^{2N}$. Vektor $\boldsymbol{m}$ se zapisuje kao $m = (m_1, m_2)$, gdje je $m_i \in \{0, \dots, q-1\}^N$.

Digitalni potpis se radi primjenom Babaijeve metode za aproksimaciju CVP problema na m upotrebom javnog ključa R i dobiva se $(s, t) \in Z^{2n}$. U praksi vektor t se ne mora pamtiti jer se jednostavno izračuna uz pomoć h.

Duljina digitalnog potpisa u NTRUSign-u je puno manja nego u ostalim digitalnim potpisima temeljenim na rešetkama, ali je ipak još uvijek znatno veća od duljine trenutnih digitalnih potpisa poput DSA[10].

Verifikacija digitalnog potpisa jednaka je kao kod GGH digitalnog potpisa. [35]

9.3. Sigurnost NTRUSign-a

U [36] Q. Nguyen i O. Regev pokazali su da NTRUSign nije siguran te da se privatni ključ može otkriti i nakon samo 400 digitalnih potpisa s istim parom ključeva metodom učenja skrivenog paralelepipeda.

Kako bi se izbjegla ovakva vrsta napada na privatni ključ u NTRUSign-u je dodana perturbacija sažetka. Još nije napravljena kriptoanaliza takvog NTRUSign-a, ali Nguyen i Regev smatraju da bi se i takav slučaj uspio probiti metodom učenja skrivenog paralelepipeda.

[10] Duljina NTRUSign digitalnog potpisa uz N=251 je 1757 bitova dok je DSA digitalni potpis uz upotrebu SHA-1 dugačak 320 bitova

10. Ostvarenje NTRU kriptosustava

NTRU kriptosustav implementiran je u *Microsoft Visual Studio .NET 2010* alatu koristeći programski jezik *C# 3.5 i C# 4.0*.

Implementacija je podijeljena u tri dijela:

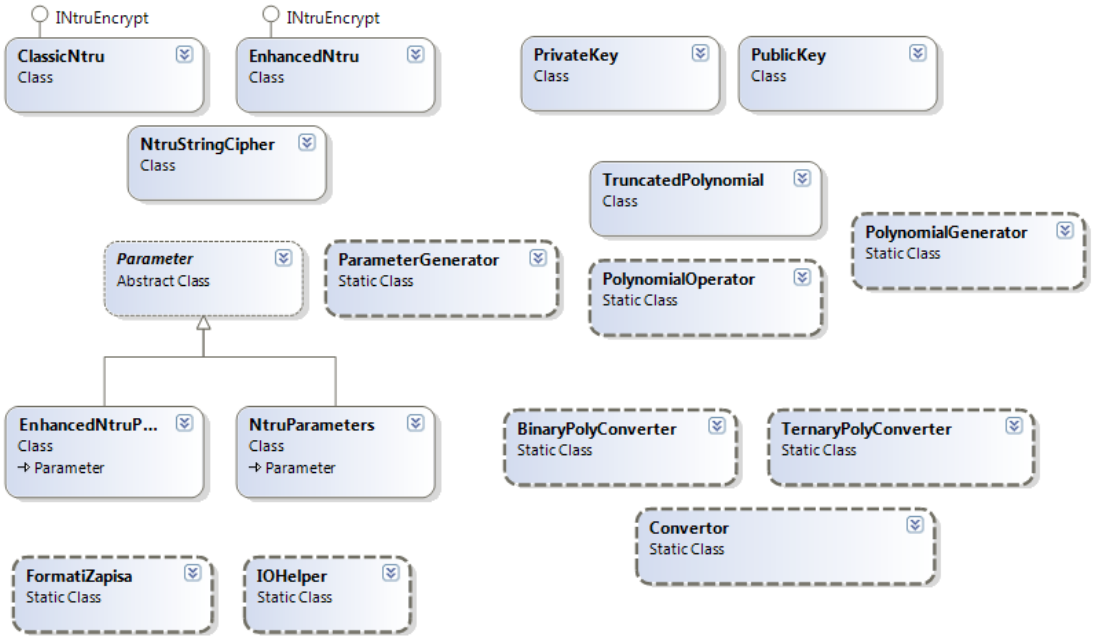
- NTRU.GUI (*.NET Framework Windows Forms 3.5*)
- NTRU.Test (*.NET Framework 4.0 Console Application*)
- NTRU (*.NET Framework 3.5 Class Library*)

U *3.5* tehnologiji napravljeno je korisničko grafičko sučelje za jednostavnu upotrebu NTRU kriptosustava i usporedbu s RSA kriptosustavom.

U *4.0* tehnologiji napravljena je konzolna aplikacija za testiranje pojedinih komponenti kriptosustava. Prilikom testiranja izvodilo se preko milijun iteracija nekog od postupaka što je korištenjem paralelizma unutar *C# 4.0* uvelike ubrzano.

Aplikacija u *4.0* tehnologiji zahtijeva instalaciju *.NET 4.0 frameworka*, dok aplikacija s korisničkim grafičkim sučeljem ne zahtijeva nikakve prethodne pripreme.

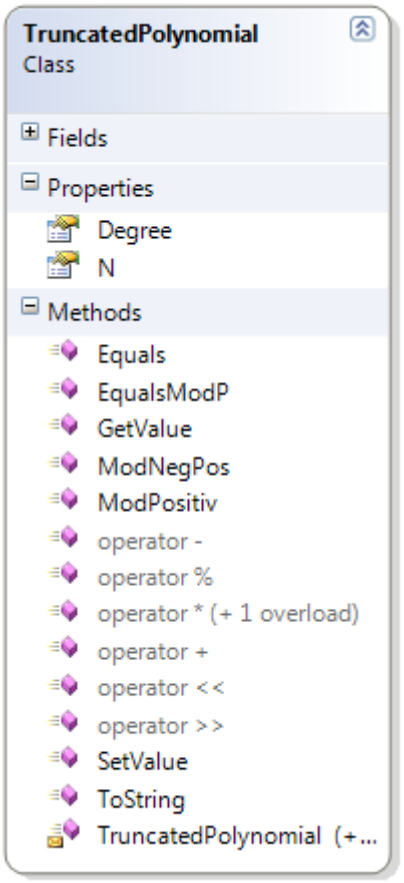
U NTRU biblioteci (engl. *class library*) implementirani su svi objekti i matematička logika potrebna za rad NTRU kriptosustava. Na slici 10.1 prikazan je dijagram klasa cijele biblioteke, a važniji dijelovi opisani su u poglavljima u nastavku rada.



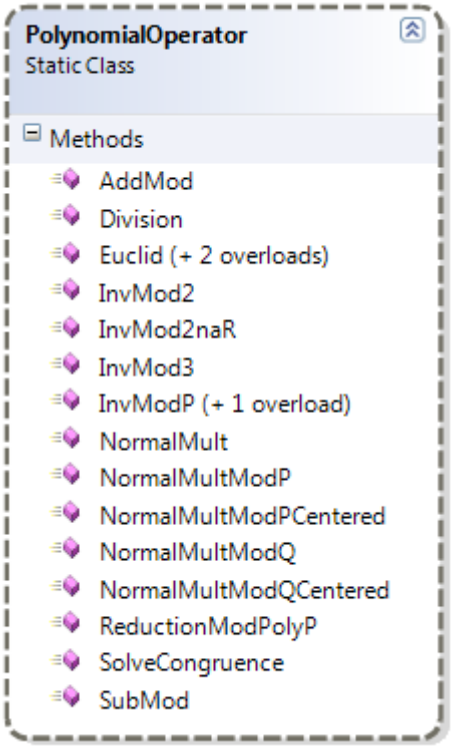
Slika 10.1 NTRU dijagram klasa

10.1. Reprezentacija polinoma i operacije nad polinomima

Polinom je definiran u klasi `TruncatedPolynomial` koja u sebi sadržava duljinu polinoma i listu koeficijenata polinoma. Na primjer, ako je neki polinom iz prstena $Z_3[X]/(X^6 - 1)$ jednak $a(X) = -1 + X^2 + X^3 - X^4$, tada je njegova reprezentacija jednaka $[-1,0,1,1,-1,0]$. Klasa `TruncatedPolynomial` sa svim svojstvima i metodama prikazana je na slici 10.2.



Slika 10.2 Klasa TruncatedPolynomial



Slika 10.3 Klasa s polinomnim operatorima

U statičkoj klasi `PolynomialOperator` implementirani su svi operatori nad polinomima koji se upotrebljavaju u NTRU kriptosustavu. Važniji operatori opisani su u sljedećim poglavljima, a sve metode u klasi prikazane su na slici 10.3.

10.1.1. Konvolucijsko množenje polinoma

Vremenski najzahtjevnija operacija u NTRU kriptosustavu je konvolucijsko množenje, složenosti N^2.

U relaciji (10.1) ponovljena je formula za konvolucijsko množenje iz poglavlja 3.2.3 koja je implementirana u algoritmu prikazanom na slici 10.4:

$$r(x) = p(x) \circledast q(x) = r_0 + r_1 x + r_2 x^2 + \cdots + r_{N-1} x^{N-1} + r_N x^N, uz$$
$$r_k = a_0 b_k + a_1 b_{k-1} + \cdots + a_k b_0 + a_{k+1} b_N + a_{k+2} b_{N-1} + \cdots + a_N b_{k+1} \quad (10.1)$$

```
Ulazni parametri: a(x) i b(x), polinomi; N, duljina polinoma
Izlazni parametri: polinom c(x) = a(x) * b(x)
1: c(x) = 0
2: for k = 0 to N - 1
3:     koef = 0
4:     for i = 0 to k
5:           koef = koef + a_i * b_k-i
6:     for i = k + 1 to N - 1
7:           koef = koef + a_i * b_k-i+N
8:     c_k = koef
9: return c
```

Slika 10.4 Pseudokod konvolucijskog množenja polinoma

10.1.2. Redukcija polinoma

Redukcija polinoma modulo p je zapravo redukcija koeficijenata polinoma modulo p.

Budući da se nakon izvođenja operacija nad polinomima često radi redukcija polinoma, redukcija je implementirana unutar operatora zbrajanja, oduzimanja i množenja. Na primjer, prilikom množenja se u koraku 8 algoritma na slici 10.4 varijabla `koef` prije pridruživanja reducira modulo p.

Modularna redukcija se računa dijeljenjem, odnosno gleda se ostatak prilikom dijeljenja. No, za module koji su potencija broja 2 reduciranje se može znatno ubrzati jer vrijedi sljedeće:

$$a \ (mod\ 2^r) = a \ \& \ (2^r - 1). \quad (10.2)$$

Dakle, umjesto dijeljenja radi se logička operacija AND. Ovakav način redukcije koristi se kada je veliki modul jednak 2^r i ubrzava reduciranje i do 40%.

10.1.3. Inverz polinoma

Prilikom generiranja ključeva računaju se inverzi polinoma modulo p i q. U [37] dani su algoritmi za efikasno računanje inverza polinoma modulo 2, modulo 3 i modulo p^r gdje je p prost broj.

Ulazni parametri: polinom $a(X) \in Z_3[X]/(X^N - 1)$; duljina polinoma N

Izlazni parametri: polinom $a(X)^{-1}\ (mod\ 3)$

```
 1:  b(x) = 1, c(x) = 0, g(x) = x^N-1, f(x) = a(x)
 2:  k = 0
 3:  dok je(1)
 4:     ako stupanj(f)<0
 5:        vrati „NEMA INVERZA!"
 6:     dok je(f_0 == 0)
 7:        f = f/X
 8:        c = c*X
 9:        k = k + 1
10:     ako stupanj(f) == 0 && f_0 != 0
11:        prekid petlje
12:     ako stupanj(f)<stupanj(g)
13:        zamijeni b i c
14:        zamijeni f i g
15:     ako f_0 (mod 3) == g_0 (mod 3)
16:        f = f - g (mod 3)
17:        b = b - c (mod 3)
18:     inače
19:        f = f + g (mod 3)
20:        b = b + c (mod 3)
21:  b = b*X^N-k
22:  ako f_0 (mod 3) == 1
23:     vrati b
24:  inače
25:     vrati (-1) * b
```

Slika 10.5 Algoritam računanja inverza polinoma modulo 3

Ulazni parametri: polinom $a(X) \in Z_2[X]/(X^N - 1)$; duljina polinoma N

Izlazni parametri: polinom $a(X)^{-1}\ (mod\ 2)$

```
 1:  b(x) = 1, c(x) = 0, g(x) = x^N-1, f(x) = a(x)
 2:  k = 0
 3:  dok je(1)
 4:     ako stupanj(f)<0
 5:        vrati „NEMA INVERZA!"
 6:     dok je(f_0 == 0)
 7:        f = f/X (rotacija koeficijenata za 1 mjesto ulijevo)
 8:        c = c*X (rotacija koeficijenata za 1 mjesto udesno)
 9:        k = k + 1
10:     ako f(x) == 1
11:        vrati b*X^(N-k)(rotacija koeficijenata za k mjesta ulijevo)
12:     ako stupanj(f)<stupanj(g)
13:        zamijeni b i c
14:        zamijeni f i g
15:     f = f + g (mod 2)
16:     b = b + c (mod 2)
```

Slika 10.6 Algoritam računanja inverza polinoma modulo 2

Ulazni parametri: polinom $a(X) \in Z_p[X]/(X^N - 1)$; duljina polinoma N; prost broj p; prirodni broj r

Izlazni parametri: polinom $a(X)^{-1}\ (mod\ p^r)$

```
1: b(x) = a^-1 (mod p)
2: q = p
3: dok je q < p^r
4:    q = q^2
5:    b(x) = b(x) * ( 2 - a(x)*b(x) ) (mod q)
6: vrati b(x)
```

Slika 10.7 Računanje inverza polinoma modulo p^r

Za NTRU čiji su parametri $p = 2$ i $q = prostBroj$ potrebno je definirati općeniti algoritam za računanje inverza modulo neki prost broj p.

Neki polinom $a(X) \in Z_p[X]/(X^N - 1)$ ima inverz modulo p, ako je najveći zajednički djelitelj polinoma $a(X)$ i polinoma $X^N - 1$ u prstenu $Z_p[X]$ stupnja 0.

Za računanje inverza polinoma koristi se algoritam dijeljenja polinoma i prošireni Euklidov algoritam za nalaženje najvećeg zajedničkog djelitelja polinoma.

U algoritmima na slikama 10.8 i 10.9 definiran je prošireni Euklidov algoritam i algoritam za računanje inverza $a^{-1}\ (mod\ p)$ u skupu Z.

```
Ulazni parametri: a i b, cijeli brojevi
Izlazni parametri: cijeli brojevi u, v, d za koje vrijedi a*u+b*v=d

 1:  u = 1
 2:  v = 0
 3:  d = a
 4:  t = 0
 5:  s = 1
 6:   w = b
 7:   dok je w>0
 8:    q = d/w
 9:    temp = u
10:     u = t
11:     t = temp - q*t
12:     temp = v
13:     v = s
14:     s = temp - q*s
15:     temp = d
16:     d = w
17:     w = temp - q*w
18:  vrati u,v,d
```

Slika 10.8 Prošireni Euklidov algoritam za cijele brojeve (IntEuclid)

```
Ulazni parametri: a i p, cijeli brojevi
Izlazni parametri: a^-1 (mod p)

1: odredi u,v,d za koje vrijedi a*u+p*v = d pomoću IntEuclid
2: ako d>1
3:    vrati „INVERZ NE POSTOJI"
4: inače
5:    vrati u
```

Slika 10.9 Računanje inverza modulo p u skupu Z (IntInvModP)

Na isti princip funkcionira i inverz polinoma. Za računanje inverza potrebno je izračunati polinome u,v,d za koje vrijedi $a * u + p * v = nzd(a, p) = d$.

Za potrebe Euklidovog algoritma s polinomima potrebno je definirati algoritam za dijeljenje polinoma. Operacije u algoritmu se izvode u prstenu $Z_p[X]$, a ne u prstenu $Z_p[X]/(X^N - 1)$ kako je da sada bio slučaj. Dakle, nema redukcije potencije od X.

U nastavku su prikazani: algoritam za dijeljenje polinoma (Slika 10.10), prošireni Euklidov algoritam za polinome (Slika 10.11) te na kraju algoritam za računanje inverza polinoma modulo neki prost broj p (Slika 10.12). navedeni algoritmi moraju se koristiti kada se rabi NTRU s binarnim polinomima.

Ulazni parametri: prost broj p; polinom $a(X) \in Z_p[X]$; polinom $b(X) \in Z_p[X]$ stupnja N-1 ($b_N \neq 0$)

Izlazni parametri: polinomi $q(X), r(X) \in Z_p[X]$ za koje vrijedi $a(X) = b(X) * q(X) + r(X)$ uz stupanj(r)<stupanj(b)

```
 1:  r = a, q = 0
 2:  u = b_N^-1 mod p (algoritmom IntInvModP)
 3:  d = stupanj(r)
 4:  dok je d >= N
 5:     v = u*r_d*X^(d-N)
 6:     r = r - v*b
 7:     d = stupanj(r)
 8:     q = q + v
 9:  vrati q,r
```

Slika 10.10 Dijeljenje polinoma u $Z_p[X]$ (PolyDiv)

Ulazni parametri: prost broj p; polinomi $a(X), b(X) \in Z_p[X]$ koji nisu oba 0

Izlazni parametri: polinomi $u(X), v(X), d(X) \in Z_p[X]$ za koje vrijedi $d(X) = NZD(a(X), b(X))$ i $a(X) * u(X) + b(X) * v(X) = d(X)$

```
 1:  ako b(x) == 0
 2:    vrati (1,0,a)
 3:  u = 1
 4:  d = a
 5:   v1 = 0
 6:   v3 = b
 7:   dok je v3 != 0
 8:     upotrijebi PolyDiv algoritam za izračun d = v3 * q + t3
 9:     t1 = u - q * v1
10:     u = v1
11:     d = v3
12:     v1 = t1
13:     v3 = t3
14:   v = (d - a*u)/b (sa PolyDiv - neće biti ostatka)
15:   vrati (u,v,d)
```

Slika 10.11 Prošireni Euklidov algoritam za polinome u $Z_p[X]$ (PolyEuclid)

Ulazni parametri: prost broj p; polinom $a(X) \in Z_p[X]/(X^N - 1)$

Izlazni parametri: polinom $b(X) = a(X)^{-1} \ (mod\ p)$

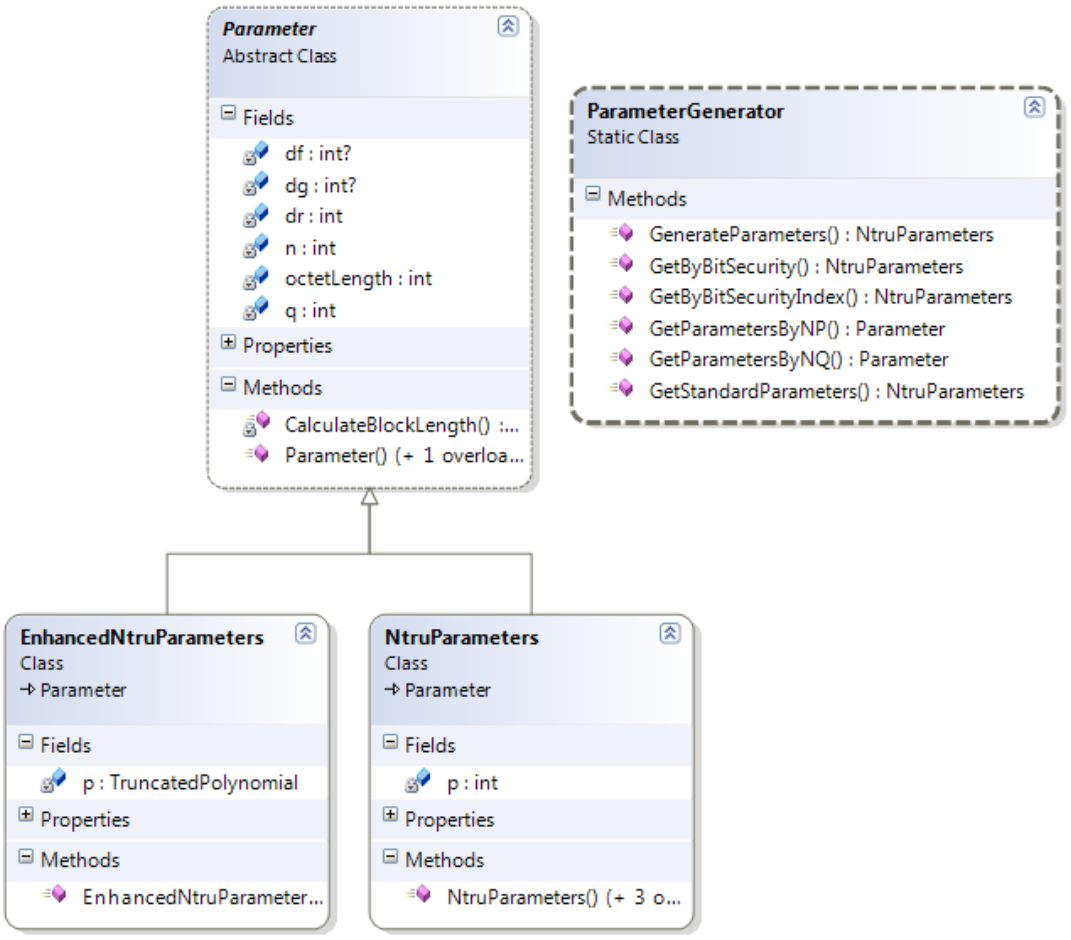
```
 1:  pokreni PolyEuclid s parametrima a, (X^N-1). Neka su (u,v,d)
     rezultati iz PolyEuclid za koje vrijedi a*u+(X^N-1)*v = d
2:  ako stupanj(d) = 0
 3:     vrati b = d^-1 (mod p) * u
 4:  inače
 5:      vrati „NEMA INVERZA"
```

Slika 10.12 Inverz polinoma modulo p (InvPolyModP)

U koraku 3 vrijednost $d^{-1} (\mathrm{mod}\ p)$ se računa algoritmom `IntInvModP` (Slika 10.9) budući da je polinom `d` zapravo konstanta.

10.2. NTRU parametri

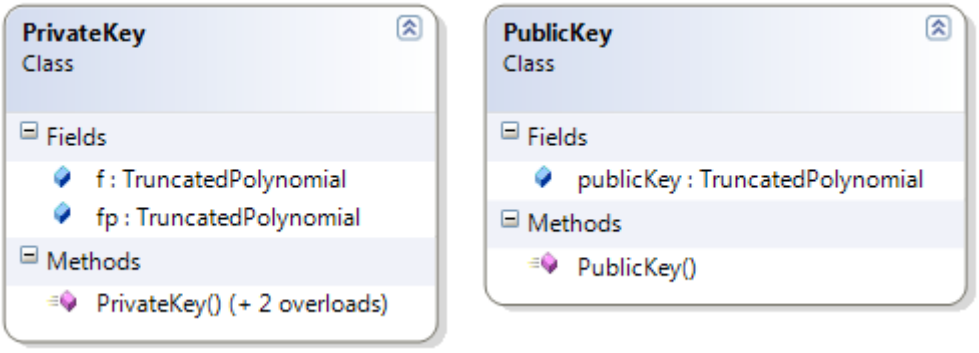
NTRU parametri su definirani u `NtruParameters` i `EnhancedNtruParameters` klasama koje su izvedene iz apstraktne klase `Parameter`. Klase u sebi sadržavaju sve NTRU parametre: N, p, q, df, dg, dr. U klasi `EnhancedNtruParameters` parametar p je polinom (objekt tipa `TruncatedPolynomial`), a u klasi `NtruParameters` p je integer (2 ili 3). U statičkoj klasi `ParameterGenerator` implementirane su metode za lakše dohvaćanje parametara prema razini sigurnosti ili prema nekim drugim svojstvima. Navedene klase prikazane su na slici 10.13.



Slika 10.13 Dijagrami klasa vezani uz parametre NTRU kriptosustava

10.3. NTRU ključevi

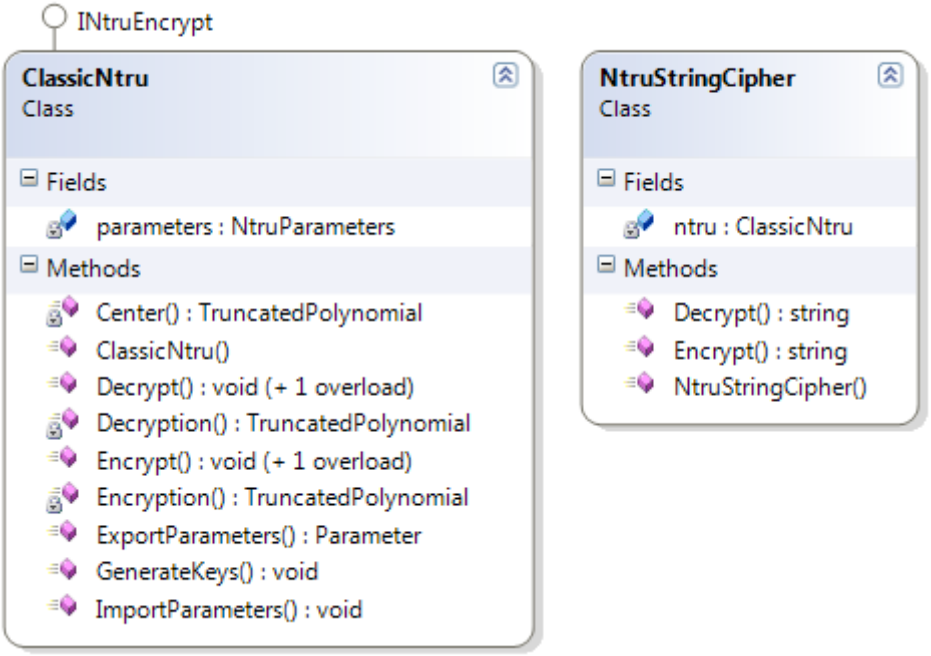
NTRU ključevi implementirani su u klasama `PrivateKey` i `PublicKey` (**Slika 10.14**).



Slika 10.14 Klase s NTRU ključevima

10.4. NTRU kriptografski algoritam

U klasi `ClassicNtru` nalazi se jezgra NTRU kriptosustava. U klasi su definirane metode za generiranje ključeva, kriptiranje i dekriptiranje. Izvorni kod klase može se pogledati u dodatku C. U klasi `NtruStringCipher` upotrebljava se `ClassicNtru` za kriptiranje i dekriptiranje tekstualnih podataka.

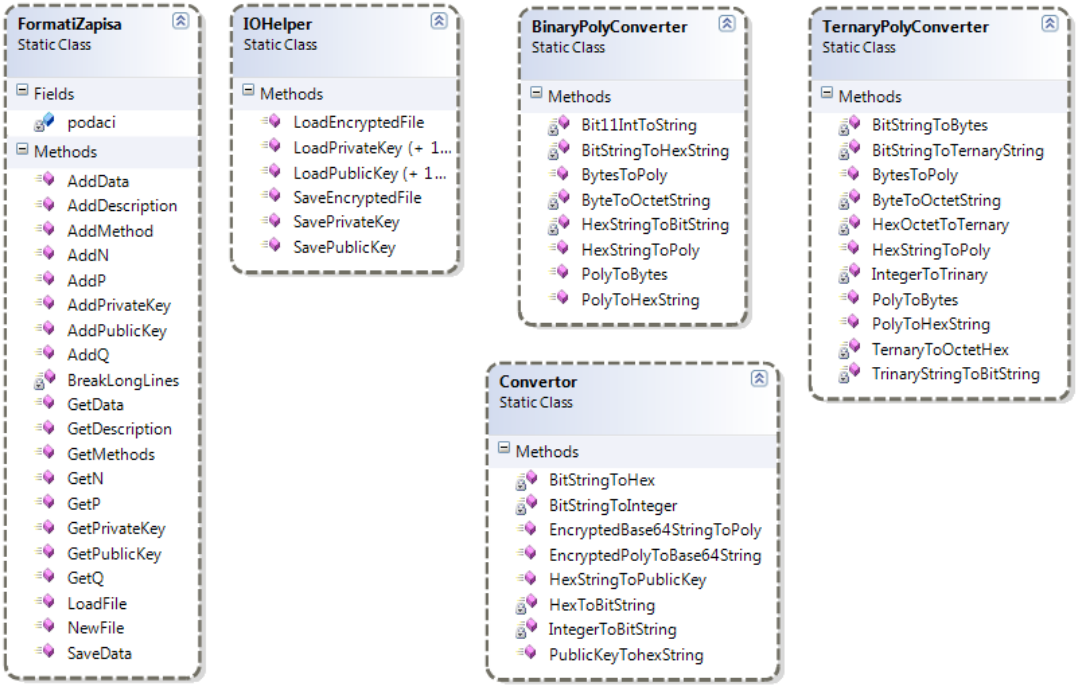


Slika 10.15 Jezgra NTRU kriptosustava

Ostale varijante NTRU-a (varijanta s $f = 1 + pF$ i varijanta s $p = 2 + X$) implementirane su u klasi `EnhancedNtru`, koja je strukturirana jednako kao i klasa `ClassicNtru`. Za te varijante kriptosustava dekriptiranje nije bilo ispravno, pa one nisu uključene u grafičko sučelje.

10.5. Pretvorbe tipova podataka

Sve pretvorbe iz binarnog oblika u polinomni oblik i obrnuto osmišljene su samostalno na temelju NAEP/SVES-3 enkripcijske sheme opisane u [27]. Klase korištene za pretvorbe prikazane su na slici 10.16.



Slika 10.16 Klase vezane uz konverzije polinoma i rad s datotekama

10.5.1. Pretvorbe čistog teksta za NTRU s binarnim polinomima

Za NTRU koji upotrebljava binarne polinome pretvorba čistog teksta iz binarne reprezentacije u polinomnu reprezentaciju je intuitivna. Svaki bit čistog teksta pretvara se u jedan koeficijent polinoma. Kako bi se upotpunio broj koeficijenata na

potrebnu duljinu dodaju se koeficijenti vrijednosti 0, a u zadnjih $\lceil \log_2(N) \rceil$ koeficijenata se zapisuje broj pridodanih koeficijenata u binarnoj reprezentaciji.

Primjer:

Neka je izvorni tekst jednak $m = [0100\ 1110\ 0011\ 0100]$ i neka je duljina polinoma jednaka $N = 23$. Polinomna reprezentacija poruke m biti će:

$$M = [0\ 1\ 0\ 0\ 1\ 1\ 1\ 0\ 0\ 0\ 1\ 1\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0].$$

Polinom je upotpunjen s dva koeficijenta vrijednosti 0 (crvena boja). Broj dodanih koeficijenata zapisan je u zadnjih $\lceil \log_2(N) \rceil = 5$ koeficijenata polinoma (plava boja).

Obrnutim postupkom pretvara se polinomna reprezentacija čistog teksta (ili dekriptiranog teksta) u binarni oblik, što se neće posebno objašnjavati.

10.5.2. Pretvorbe čistog teksta za NTRU s ternarnim polinomima

Za NTRU koji upotrebljava ternarne polinome radi se pretvorba čistog teksta iz binarne reprezentacije u polinomnu reprezentaciju s koeficijentima modulo 3. U tu svrhu izvodi se zamjena tri binarne znamenke s dvije ternarne znamenke kako slijedi:

$$\begin{aligned}
\{0,0,0\} &\rightarrow \{0,0\} \\
\{0,0,1\} &\rightarrow \{0,1\} \\
\{0,1,0\} &\rightarrow \{0,-1\} \\
\{0,1,1\} &\rightarrow \{1,0\} \\
\{1,0,0\} &\rightarrow \{1,1\} \\
\{1,0,1\} &\rightarrow \{1,-1\} \\
\{1,1,0\} &\rightarrow \{-1,0\} \\
\{1,1,1\} &\rightarrow \{-1,1\}
\end{aligned} \tag{10.3}$$

Niz bajtova čija duljina ne prelazi maksimalnu duljinu se pretvori u znakovni niz bitova, `BitString`. Na primjer niz $[34,57] = [00100010, 00111001]$ se pretvori u znakovni niz '001000100011100100010001'.

Nakon toga uzimaju se tri po tri znaka iz `BitStringa` i prema (10.3) pretvaraju se u ternarne znakove, od kojih svaki predstavlja jedan koeficijent polinoma. Ukoliko broj znakova u `BitStringu` nije višekratnik broja 3 na početak `BitStringa` se dodaje onoliko znakova '0' koliko je potrebno da se upotpuni količina do višekratnika

broja 3. Broj znakova '0' koje smo dodali na početak `BitStringa` biti će zapisan u prvom koeficijentu polinoma. Svi ostali koeficijenti polinoma koji su ostali nepopunjeni, upotpunjuju se brojevima 0, osim zadnjih $\lceil \log_3(N-1) \rceil$ koeficijenata u kojima se zapisuje koliko je znamenki 0 dodano u polinom u ternarnom obliku.

Maksimalna duljina niza bitova koji se mogu konvertirati u jedan polinom jednaka je $\frac{(N-1-\lceil \log_3(N-1) \rceil) \cdot 3}{2}$, odnosno maksimalni broj bajtova jednak je $\frac{(N-1-\lceil \log_3(N-1) \rceil) \cdot 3}{16}$. Iako asimetrični kriptosustavi ne služe kriptiranju blokova, za demonstraciju NTRU-a dopušteno je kriptiranje većih količina podataka koji će se odvajati u blokove.

Primjer:

Neka je čisti tekst $m = [1010\ 0100\ 0100\ 1111]$, a duljina polinoma $N = 23$. Ulazni tekst ima 16 bitova i niz bitova se treba nadopuniti s dva bita da bi ukupni broj bitova bio višekratnik broja 3. Dobiva se $m' = [001\ 010\ 010\ 001\ 001\ 111]$. Reprezentativni ternarni polinom je tada jednak:

$$M = [-1\ 0\ 1\ 0\ -1\ 0\ -1\ 0\ 1\ 0\ 1\ -1\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ -1\ 1].$$

Prvi koeficijent je -1 jer je broj dodanih bitova jednak 2: $-1(mod\ 3) = 2$. Koeficijenti označeni zelenom bojom su ternarna reprezentacija poruke m'. Crveni koeficijetni su nadopuna polinoma, dok je u zadnja 3 koeficijenta zapisan broj pridodanih (crvenih) koeficijenata: $021_3 = 2 \cdot 3 + 1 = 7_{10}$

Pretvorba iz polinoma u niz bajtova je obrnuti postupak. Iz zadnjih $\lceil \log_3(N-1) \rceil$ koeficijenata polinoma odredi se broj koeficijenata polinoma koji se zanemaruju. Niz preostalih koeficijenata (zanemarivši prvi koeficijent) pretvori se u niz bitova prema relaciji (10.3). Iz prvog koeficijenta se očita koliko prvih znamenaka se treba zanemariti i dobiveni `BitString` se pretvara u niz bajtova.

10.5.3. Pretvorbe javnog ključa i kriptiranog teksta

Javni ključ i kriptirani tekst su polinomi s koeficijentima modulo q. Njihova pretvorba u binarni zapis je intuitivna. Svaki koeficijent se pretvara u $\lceil \log_2(q) \rceil$ bitnu reprezentaciju broja. Dobiveni niz bitova se upotpunjuje bitovima vrijednosti 0, kako bi ukupna duljina bitova bila višekratnik broja 8, odnosno da se bitovi mogu zapisati u oktete. U zadnja 3 bita zapisat će se broj dodanih nula radi popunjavanja.

Primjer:

Neka je polinom jednak: $h = [14, 11, 27, 24]$ i neka je veliki modul jednak $q = 32$.

Tada će se koeficijenti polinoma zapisati kao niz $h = [01110\ 01011\ 11011\ 11000]$.

Dobiveni niz ima 20 bitova. Potrebno je dodati još 1 bit kako bi se dopunio broj bitova na višekratnik broja 8, uzimajući u obzir da se na kraj uvijek dodaju 3 bita sa zapisom broja dodanih bitova. Zadnja tri bita će biti 001 jer je pridodan jedan bit za popunjenje broja bitova. Dakle, rezultat će izgledati ovako:

$$h = [0111\ 0010\ 1111\ 0111\ 1000\ 0001]$$

10.5.4. Pretvorbe privatnog ključa

Pretvorba privatnog ključa za NTRU s binarnim polinomima radi se jednako kao pretvorba čistog teksta za NTRU s binarnim polinomima.

Pretvorba privatnog ključa kod NTRU-a s ternarnim polinomima nešto je drugačija od pretvorbe čistog teksta s ternarnim polinomima. Naime, u (10.3) ne postoji pretvorba ternarnog niza $[-1\ -\ 1]$ u binarni niz sa tri znamenke[11].

Privatni ključ će se iz polinomne reprezentacije u binarnu pretvarati tako da se svakih 5 ternarnih znakova pretvori u 8-bitnu binarnu reprezentaciju. Koeficijenti polinoma morat će se nadopuniti kako bi ukupni broj koeficijenata bio višekratnik broja 5. Na kraj niza ternarnih znamenaka će se dakle dodati potreban broj znamenaka vrijednosti 0. Broj dodanih znamenaka će se zatim zapisati u zadnje dvije ternarne znamenke. Pretvorbom 5 ternarnih znamenaka u 8 binarnih dobila se najveća iskoristivost, odnosno najmanja ekspanzija.

[11] Kod pretvorbe čistog teksta to ne predstavlja problem jer se uvijek kreće od binarne reprezentacije. Dakle, ako se pri pretvorbi polinoma u binarni oblik naiđe na [-1 -1] dogodila se pogreška u dekriptiranju.

Primjer:

Neka je privatni ključ jednak: $f = [-1\ 1\ 1\ 0\ -1\ 0\ 1\ 0\ 0\ 1\ -1]$. Broj koeficijenata u polinomu f je 11. Najbliži višekratnik broja 5 je broj 15. Dakle, polinom će se nadopuniti na duljinu od 15 koeficijenata, pri čemu će u zadnja dva koeficijenta biti zapisan broj dodanih ternarnih znamenaka:

$$f' = [-1\ 1\ 1\ 0\ -1\ 0\ 1\ 0\ 0\ 1\ -1\ 0\ 0\ 0\ -1] = [2\ 1\ 1\ 0\ 2\ 0\ 1\ 0\ 0\ 1\ 2\ 0\ 0\ 0\ 2].$$

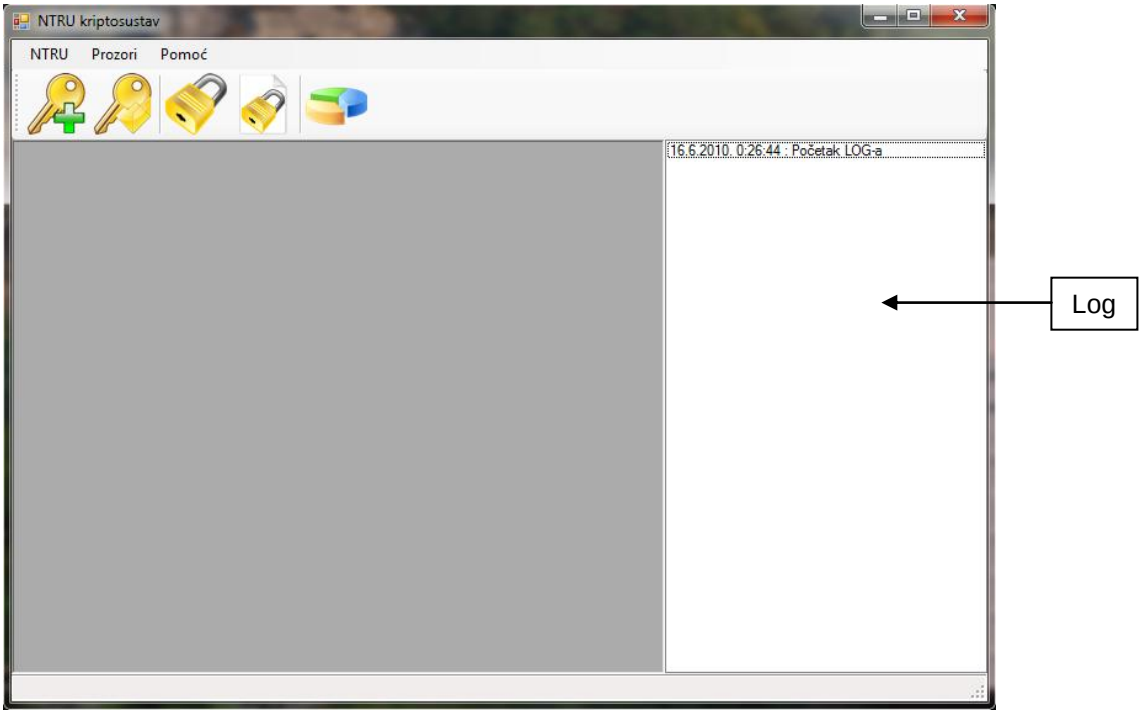
Iz prvih 5 koeficijenata dobiva se broj $2 \cdot 3^4 + 1 \cdot 3^3 + 1 \cdot 3^2 + 2 = 200$, odnosno u binarnom obliku 11001000. Na isti način dobivaju se i ostali okteti te je konačni rezultat jednak: $F = [11001000\ 00011100\ 10100100]$.

10.6. Problemi kod implementacije NTRU-a

NTRU enkripcijska shema je postala standard u prosincu 2008., no standard nije besplatan. Zbog toga implementacija nije u skladu s normama koje se propisuju u standardu. Ovdje se ponajprije misli na norme o načinu zapisivanja ključeva i ostalih parametara kriptosustava. Budući da je zadatak ovog rada bio implementirati NTRU kriptosustav, a ne cijelu shemu kriptiranja, ovaj nedostatak ne predstavlja veliki problem.

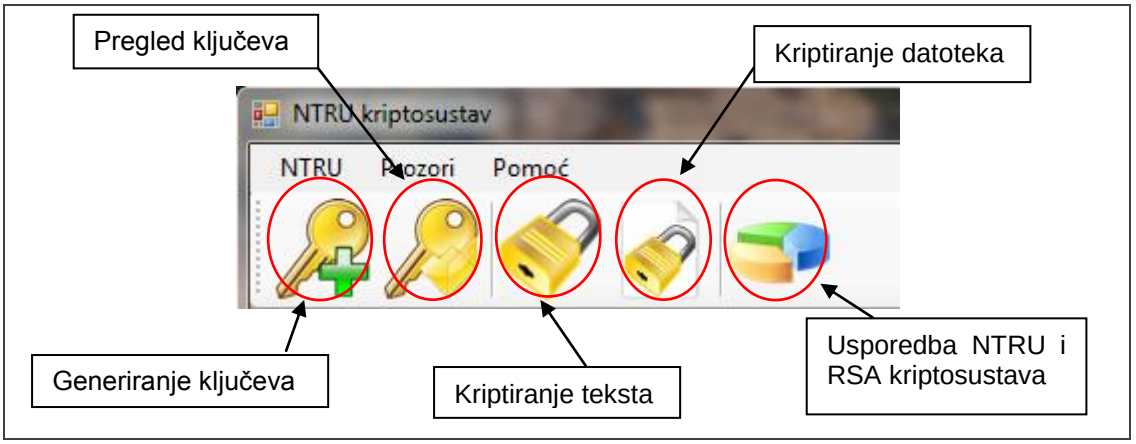
10.7. Upute za korištenje

Na slici 10.17 prikazan je glavni prozor aplikacije. Aplikacija ima alatnu traku za jednostavno pozivanje različitih funkcija unutar aplikacije. Sa desne strane nalazi se ispis loga za jednostavniji pregled informacija o radu aplikacije.



Slika 10.17 Glavni prozor aplikacije

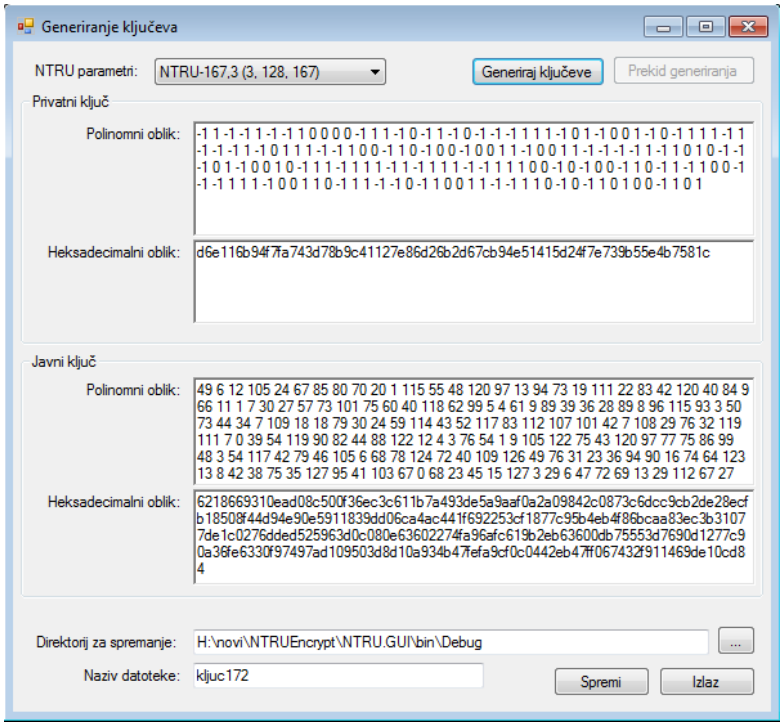
U alatnoj traci može se odabrati *Generiranje ključeva*, *Pregled ključeva*, *Kriptiranje teksta*, *Kriptiranje datoteke* i *Usporedba NTRU i RSA kriptosustava* (Slika 10.18).



Slika 10.18 Alatna traka glavnog prozora aplikacije

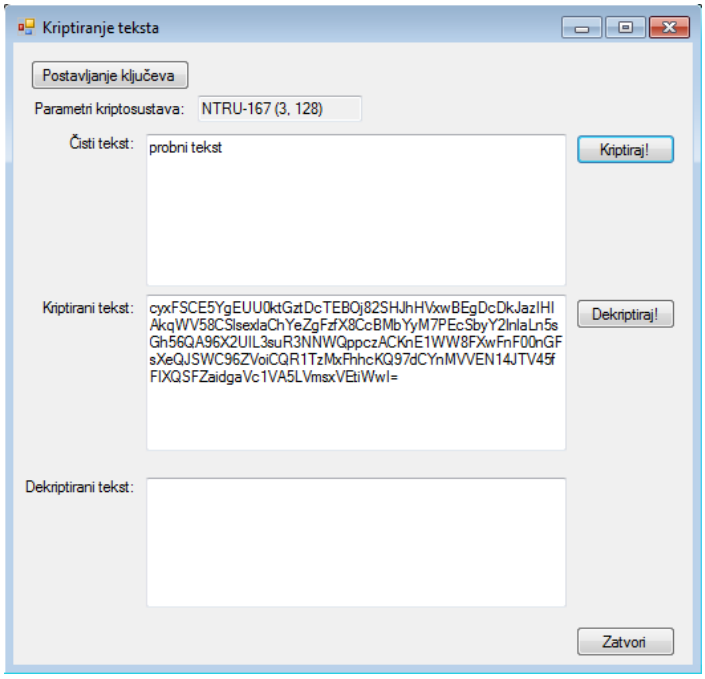
Prozor za generiranje ključeva prikazan je na slici 10.19. Nakon odabira parametara pritiskom na *Generiraj ključeve* počinje generiranje ključeva. Nakon generiranja, javni i privatni ključ se ispisuju u polinomnom obliku i u heksadecimalnom obliku. Ključevi se u heksadecimalnom obliku pohranjuju u datoteku koju je odabrao

korisnik. Javni ključ se pohranjuje u datoteku s ekstenzijom *.npu, a privatni ključ s ekstenzijom *.npr. Primjeri zapisa ključeva u datotekama nalaze se u dodatku B.



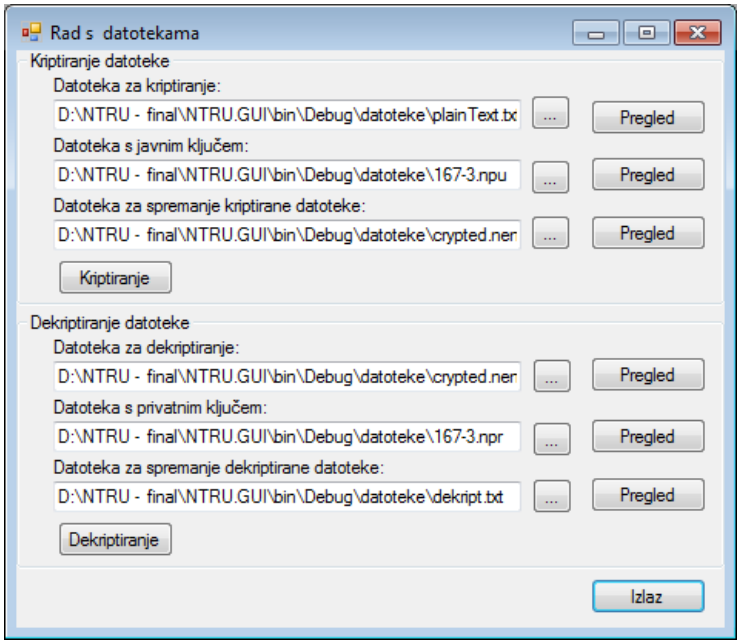
Slika 10.19 Prozor za generiranje ključeva

Prozor za kriptiranje teksta prikazan je na slici 10.20. Prije kriptiranja klikom na gumb *Postavljanje ključeva* otvara se prozor za odabir ključa, sličan prozoru za generiranje ključeva. Ključ se može učitati iz datoteke ili se može generirati novi ključ. Klikom na tipku OK prozor za odabir ključa se zatvara i svi potrebni parametri kriptosustava se učitavaju (parametri p, q i N te javni i privatni ključ). Sada se može unijeti tekst za kriptiranje (ili dekriptiranje) i klikom na gumb *Kriptiraj* (*Dekriptiraj*) izvrši se postupak kriptiranja odnosno dekriptiranja.



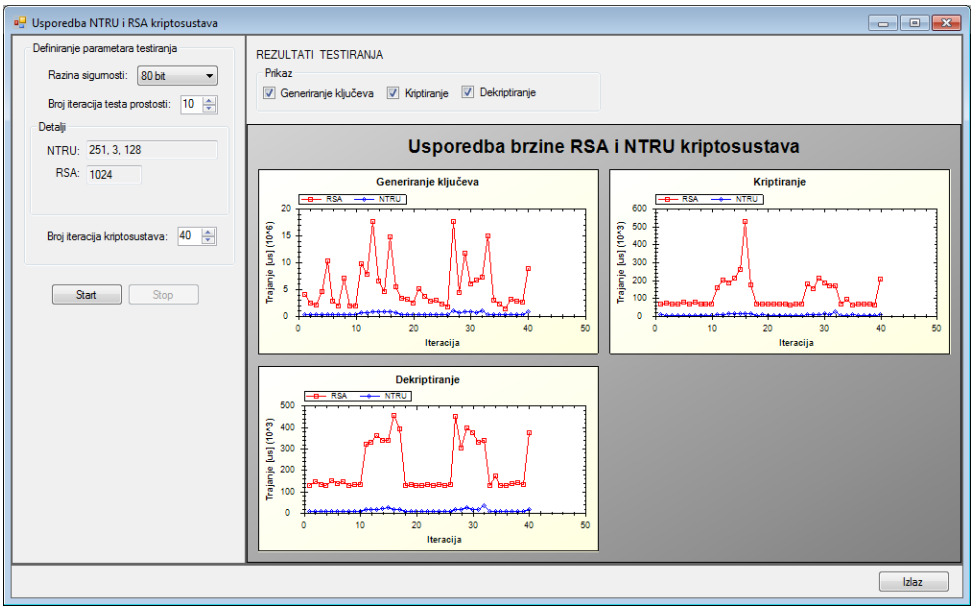
Slika 10.20 Prozor za kriptiranje teksta

Prozor za kriptiranje datoteka prikazan je na slici 10.21. Nakon odabira svih potrebnih datoteka (datoteke s ključem, datoteke s ulaznim tekstom i datoteke za spremanje izlaznog teksta) može se pokrenuti postupak kriptiranja, odnosno dekriptiranja. Pojedine datoteke se mogu pregledati pritiskom tipke *Pregled*.



Slika 10.21 Prozor za kriptiranje datoteka

Prozor za usporedbu NTRU i RSA kriptosustava prikazan je na slici 10.22. Ovdje se pokreće simulacija NTRU i RSA kriptosustava za odabranu razinu sigurnosti i sa odabranim brojem iteracija. Na kraju simulacije vrijeme kriptiranja, dekriptiranja i generiranja ključeva prikazuje se grafički. Crvenom bojom je označen RSA kriptosustav, a plavom bojom NTRU kriptosustav.



Slika 10.22 Prozor za usporednu simulaciju NTRU i RSA kriptosustava

11. Rezultati i usporedba s RSA kriptosustavom

Sve simulacije kriptosustava izvedene su na računalu s *Intel Core 2 Quadro* procesorom frekvencije 2.66 GHz.

11.1. Pogrešno dekriptiranje

Iako je s NAEP enkripcijskom shemom [31] uklonjena mogućnost probijanja NTRU kriptosustava na temelju pogreški u dekriptiranju, pogrešno dekriptiranje još uvijek predstavlja veliku manu NTRU kriptosustava.

U okviru programske implementacije NTRU-a napravljena je analiza vjerojatnosti pogrešnog dekriptiranja. Prilikom testiranja uspješnosti dekriptiranja izvođeno je 10 milijuna iteracija kriptiranja/dekriptiranja. Pri tome, generirano je 1000 različitih ključeva i napravljeno 10 000 iteracija testiranja uspješnosti dekriptiranja za svaki ključ.

Broj od 1000 ključeva se čini malim za izvođenje prihvatljivog zaključka, ali zbog ograničenih računalnih resursa nije bilo moguće raditi više iteracija.

Za potrebe testiranja napravljen je paralelni program u *.NET Framework 4.0* tehnologiji. Testiranje je trajalo tri dana, a rezultati su prikazani u tablici 11.1.

Tablica 11.1 Vjerojatnost pogrešnog dekriptiranja za preporučene parametre

NTRU – N, p	Očekivana vjerojatnost[12]	Eksperimentalna vjerojatnost
NTRU – 167, 3	$5.5 \cdot 10^{-5}$	$5.00 \cdot 10^{-5}$
NTRU – 251, 3	$1.5 \cdot 10^{-6}$	$1.87 \cdot 10^{-6}$
NTRU – 503, 3	$4.8 \cdot 10^{-5}$	$4.16 \cdot 10^{-5}$

Eksperimentalna vjerojatnost za NTRU-167 je $5 \cdot 10^{-5}$. To znači da će se na svakih 20 tisuća poruka dogoditi pogreška u dekriptiranju. Za NTRU-251 u prosijeku će tek nakon više od pola milijuna poruka doći do pogreške.

[12] Očekivana vjerojatnost je izračunata preko formule (6.7) iz poglavlja 6.1.7 Pogrešno dekriptiranje

11.2. Analiza brzine NTRU kriptosustava

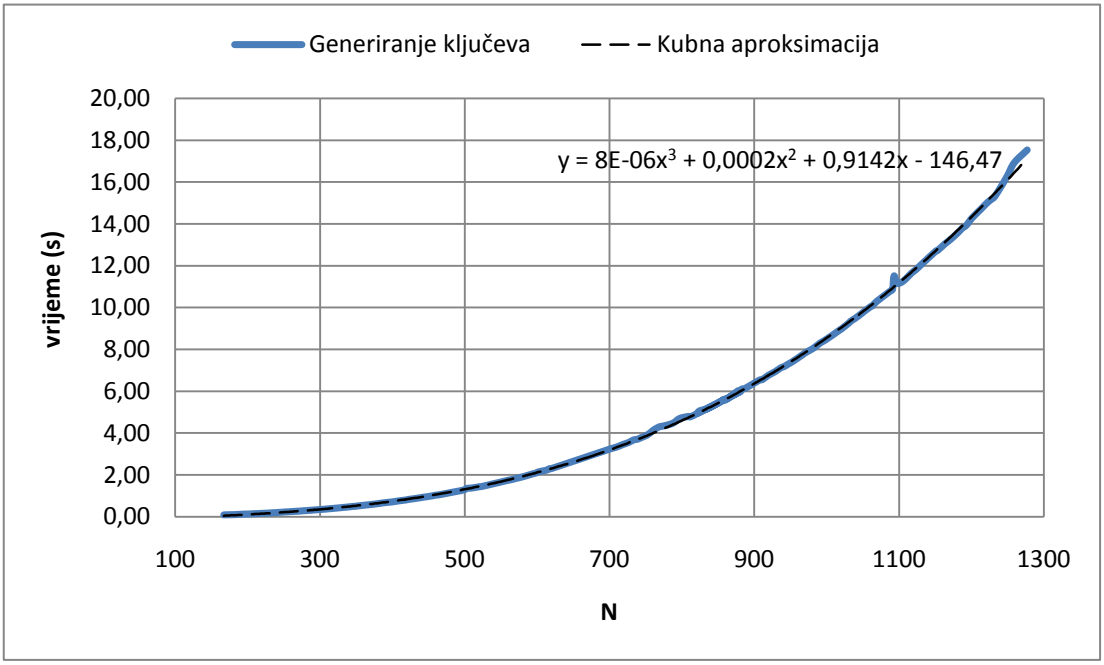
Svi rezultati, prikazani u nastavku, nastali su na temelju vlastitog ostvarenja NTRU kriptosustava.

11.2.1. NTRU s ternarnim polinomima

Na slici 11.1 prikazan je graf ovisnosti vremena generiranja ključeva o duljini[13] polinoma za ternarne polinome. Ostali parametri su bili fiksirani na vrijednostima $q = 128,\ d_f = 61, d_g = 50, d_r = 50$. Najtočnija aproksimacija dobivena je kubnom funkcijom:

$$t(N) = 8 \cdot 10^{-6} t^3 + 0.0002 t^2 + 0.9142 t - 146.47 \qquad (11.1)$$

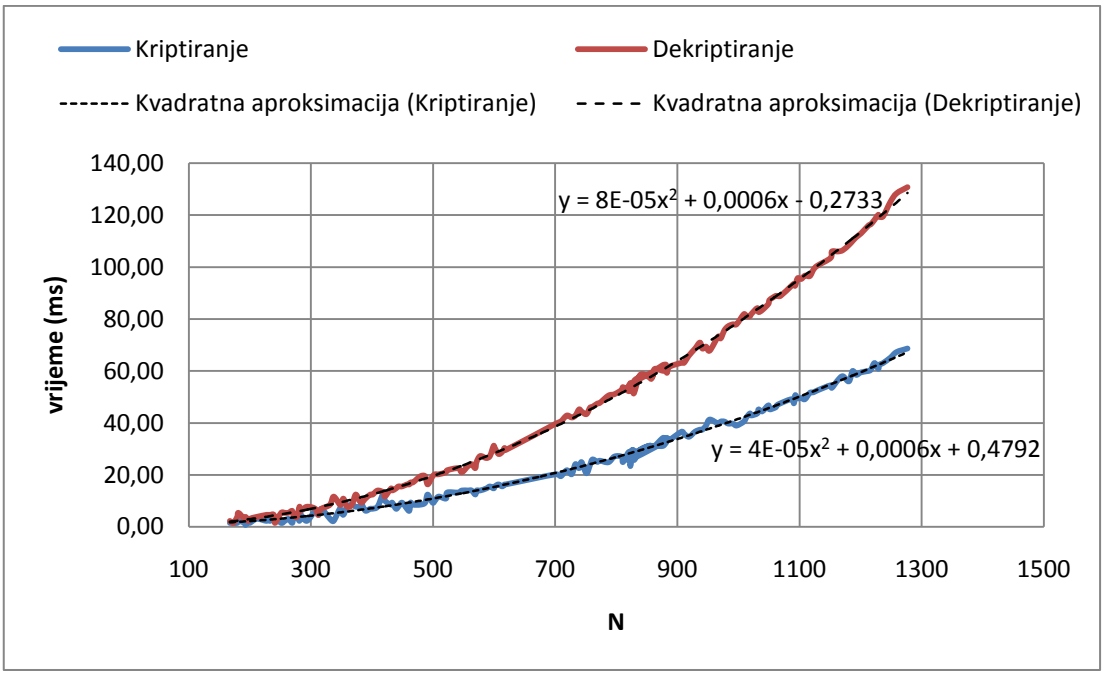
Aproksimirana kubna funkcija ima vrlo male vodeće koeficijente što ju čini sporo rastućom. U komercijalnim proizvodima se ne upotrebljavaju polinomi duljine veće od 503 i za taj interval porast vremena generiranja ključeva je zanemariv.



Slika 11.1 Brzina generiranja ključeva za NTRU s ternarnim polinomima u ovisnosti o duljini polinoma

[13] Duljine polinoma u svim mjerenjima su svi prosti brojevi iz intervala prikazanog grafovima

Vrijeme kriptiranja i dekriptiranja sporo raste s povećanjem duljine polinoma. Dekriptiranje u prosijeku traje dva puta dulje nego kriptiranje (Slika 11.2). Najbolje aproksimacije dobivene su kvadratnim funkcijama s malim vodećim koeficijentom.

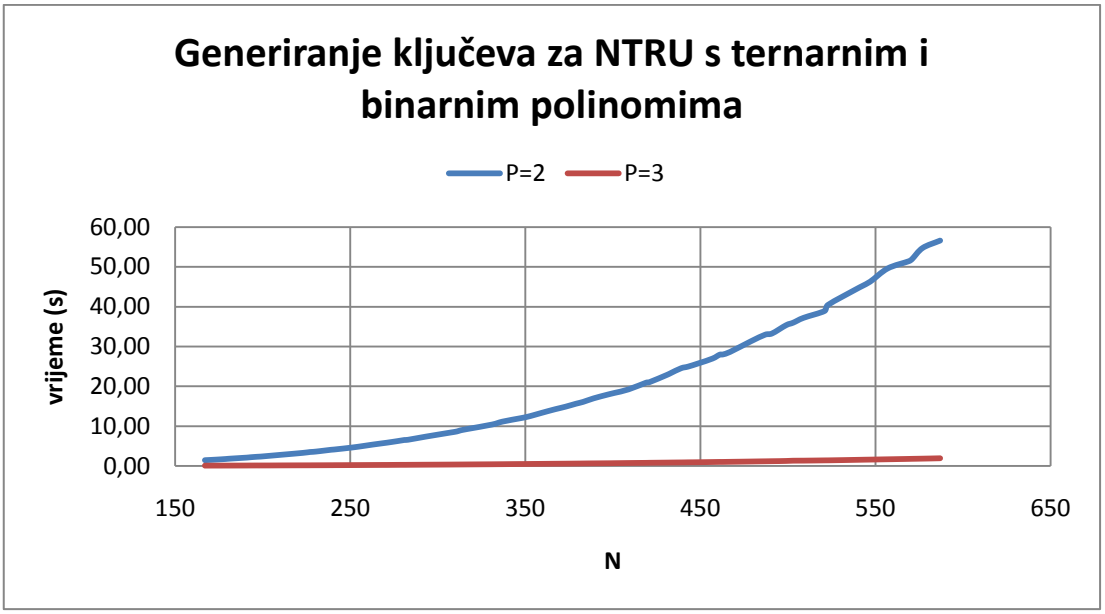


Slika 11.2 Brzina kriptiranja i dekriptiranja u ovisnosti o duljini polinoma

Parametar q nema utjecaja na brzinu kriptosustava, nego samo na duljinu kriptirane poruke i duljinu javnog ključa.

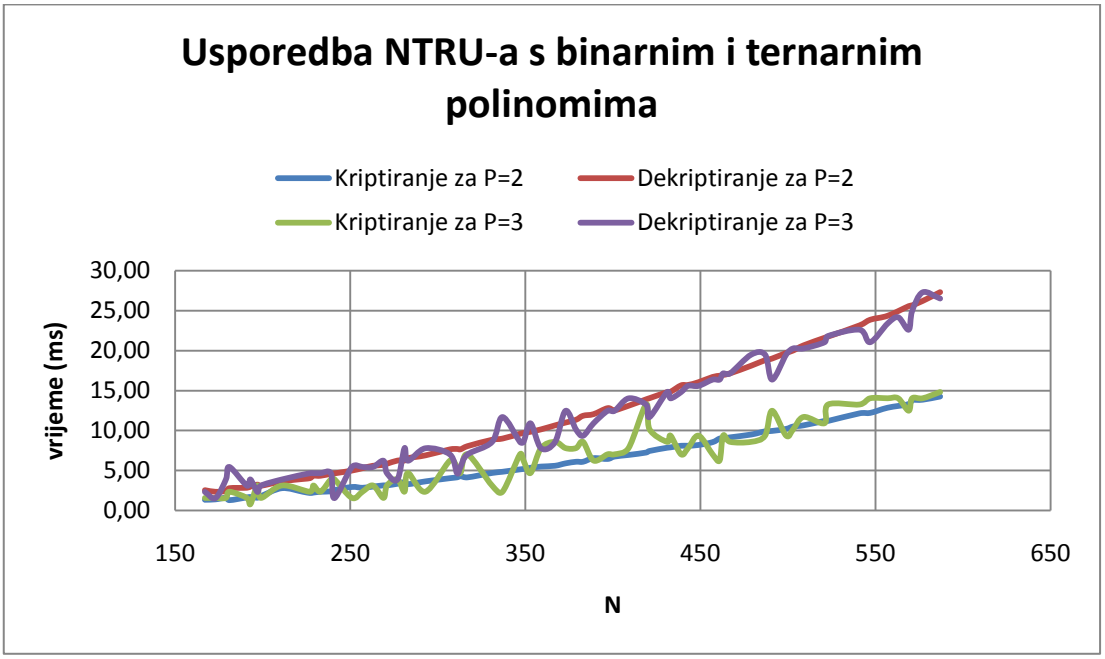
11.2.2. NTRU s binarnim polinomima

NTRU s binarnim polinomima je sporiji od NTRU-a s ternarnim polinomima. Generiranje ključeva je puno sporije zbog računanja inverza modulo prost broj. Za računanje inverza polinoma modulo 2 i 3 postoje brzi algoritmi, dok je za općeniti slučaj invertiranje veoma sporo. Na slici 11.3 prikazana je usporedba vremena generiranja ključeva za ternarne i binarne polinome.



Slika 11.3 Usporedba vremena generiranja ključeva za NTRU s ternarnim (p=3) i binarnim (p=2) polinomima

Dekriptiranje je neznatno sporije kod NTRU-a s binarnim polinomima jer se ovdje reduciranje modulo q radi uz pomoć dijeljenja, dok se kod ternarnih polinoma reduciranje radilo logičkom operacijom AND (jer je parametar q potencija broja 2).



Slika 11.4 Usporedba NTRU-a s ternarnim i binarnim polinomima pri operacijama kriptiranja i dekriptiranja

Iz grafa se može uočiti da je vrijeme kriptiranja i dekriptiranja kod NTRU-a s ternarnim polinomima dosta promjenjivo. Razlog tome je vjerojatno sljedeći:

Ako polinom ima puno koeficijenata jednakih 0, konvolucijsko množenje je brže. Kod ternarnih polinoma koeficijenti se biraju iz skupa $\{-1,0,1\}$, a kod binarnih iz skupa $\{0,1\}$. Ternarni polinomi će zato imati veću razliku između pojedinih instanci polinoma, pa stoga i veću razliku u vremenu kriptiranja (dekriptiranja).

Vrijeme kriptiranja i dekriptiranja je podjednako za obje varijante kriptosustava, ali generiranje ključeva je puno kraće s ternarnim polinomima, pa će se u daljnjim analizama rabiti takav NTRU.

11.2.3. Brzina NTRU-a s obzirom na razinu sigurnosti

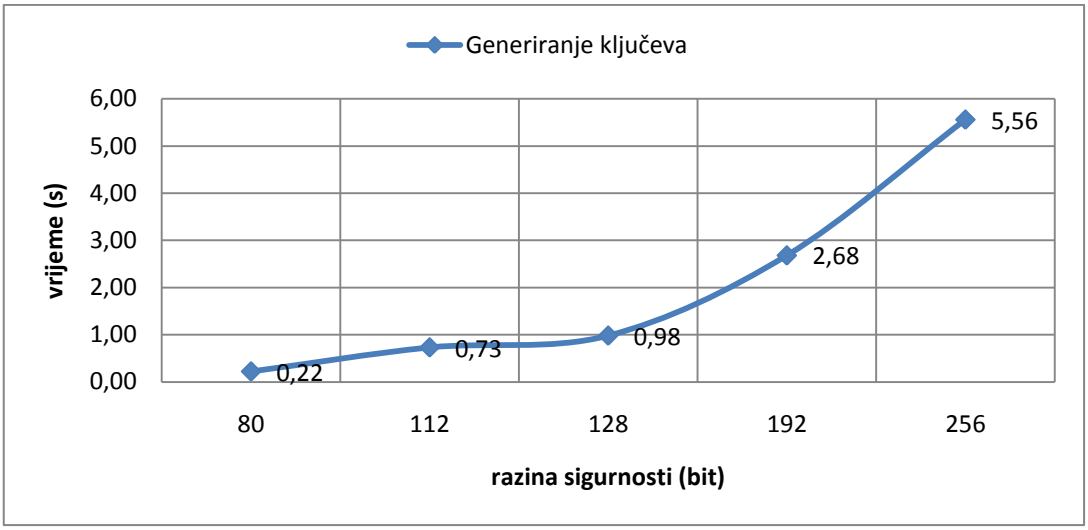
U tablici 11.2 prikazani su konzervativni[14] parametri NTRU kriptosustava s obzirom na razinu sigurnosti, preuzeti iz [27].

Tablica 11.2 Konzervativni parametri NTRU-a s obzirom na sigurnost

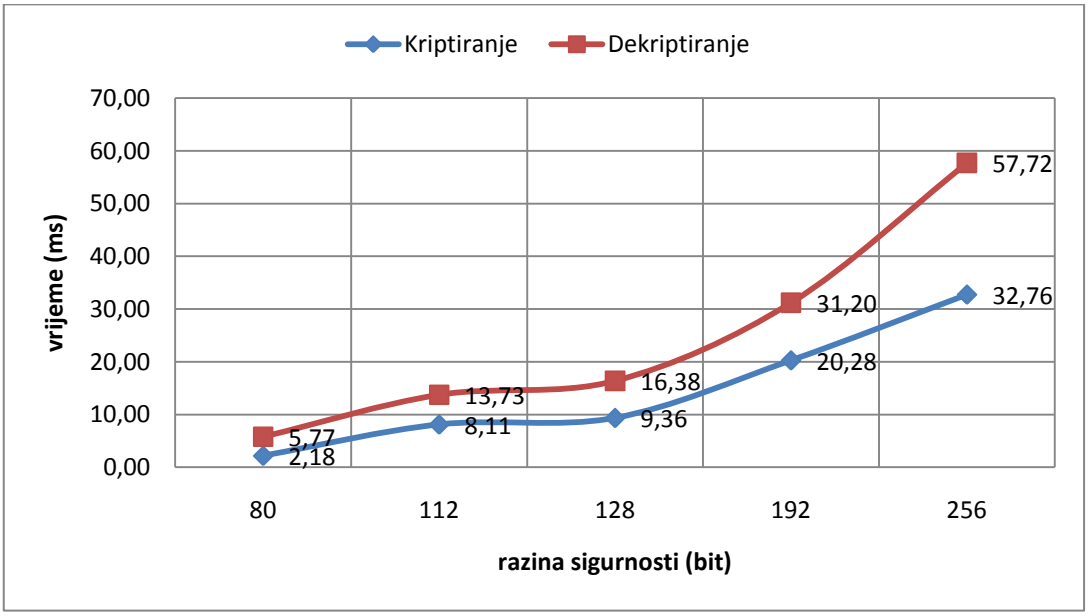
Razina sigurnosti	N	p	q	$d_f = d_r$	d_g
80	251	3	2048	50	24
112	401	3	2048	113	133
128	449	3	2048	134	149
192	653	3	2048	194	217
256	853	3	2048	268	284

Brzina NTRU kriptosustava za parametre iz tablice 11.2 prikazana je na slikama 11.5 i 11.6.

[14] Pod konzervativnim se misli na to da su stvarne razine sigurnosti veće od preporučenih zbog mogućih novih prijetnji u budućnosti. Na primjer NTRU s razinom sigurnosti 112 bitova zapravo ima sigurnost od 154 bitova s obzirom na danas poznate napade.



Slika 11.5 Graf trajanja generiranja ključeva u ovisnosti o razini sigurnosti za NTRU kriptosustav



Slika 11.6 Graf vremena kriptiranja i dekriptiranja u ovisnosti o razini sigurnosti za NTRU kriptosustav

11.3. Usporedba s RSA kriptosustavom

Prilikom implementacije RSA kriptosustava korištena je klasa `BigInteger` dostupna unutar *BouncyCastle* projekta[15].

U tablici 11.3 prikazane su duljine ključeva za pojedinu razinu sigurnosti za RSA kriptosustav.

Tablica 11.3 RSA parametri s obzirom na sigurnost

Razina sigurnosti	Duljina ključa
80	1024
112	2048
128	3072
192	7680
256	15360

Maksimalna duljina ulaznog teksta za RSA kriptosustav jednaka je duljini modula (ako se ne koristi shema kriptiranja). Duljina izlaznog teksta za RSA kriptosustav također je jednaka duljini modula, odnosno prilikom kriptiranja u prosijeku ne dolazi do ekspanzije poruke.

U tablici 11.4 za svaku razinu sigurnosti prikazana je duljina ključeva NTRU-a u bitovima. Maksimalna duljina ulaznog teksta za NTRU jednaka je duljini privatnog ključa, dok je duljina kriptiranog teksta jednaka duljini javnog ključa. Faktor ekspanzija poruke kod kriptiranja s NTRU kriptosustavom je približno jednak 6.9 za konzervativne parametre iz tablice 11.2.

Tablica 11.4 Duljina ključeva za NTRU kriptosustav s obzirom na sigurnost

Razina sigurnosti	duljina ključa (bit)		
	javni	privatni	ukupno
80	400	2768	3168
112	640	4416	5056
128	720	4944	5664
192	1040	7192	8232
256	1352	9392	10 744

[15] http://www.bouncycastle.org/

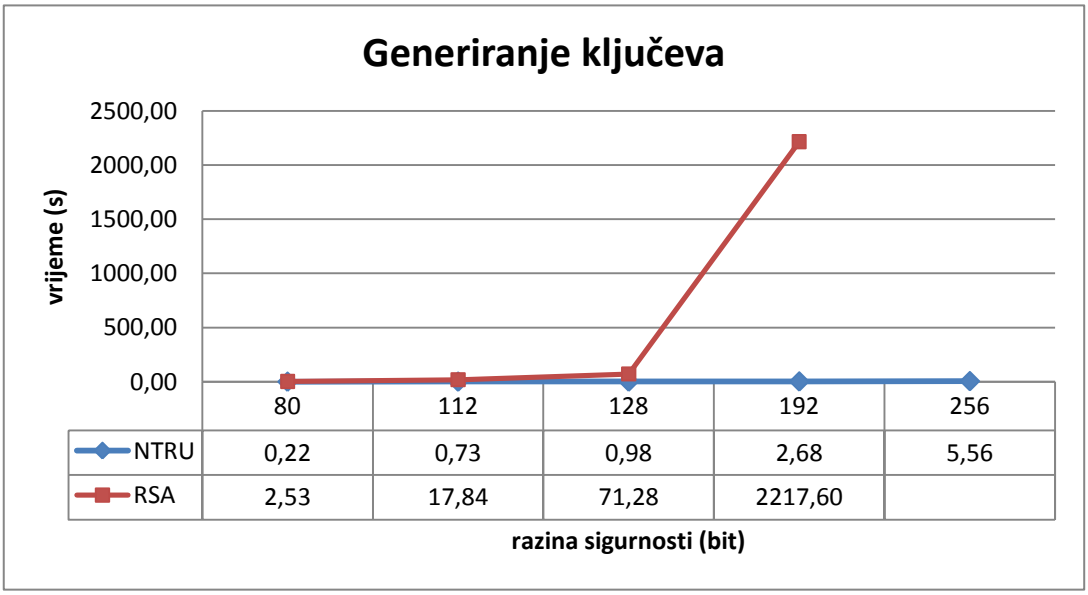
U komercijalne svrhe koriste se parametri kao u tablici 6.2 iz poglavlja 6.1.1. Za te parametre, duljina ključeva je približno dva puta manja nego za konzervativne parametre, a kriptirani tekst je prosječno 6 puta dulji od čistog teksta.

U tablici 11.5 prikazana je složenost RSA kriptosustava s obzirom na duljinu ključa i složenost NTRU kriptosustava s obzirom na duljinu polinoma. NTRU kriptosustav je puno brži u generiranju ključeva i dekriptiranju, a kako će se kasnije pokazati i u kriptiranju je NTRU puno brži od RSA kriptosustava.

Tablica 11.5 Usporedba složenosti NTRU i RSA kriptosustava

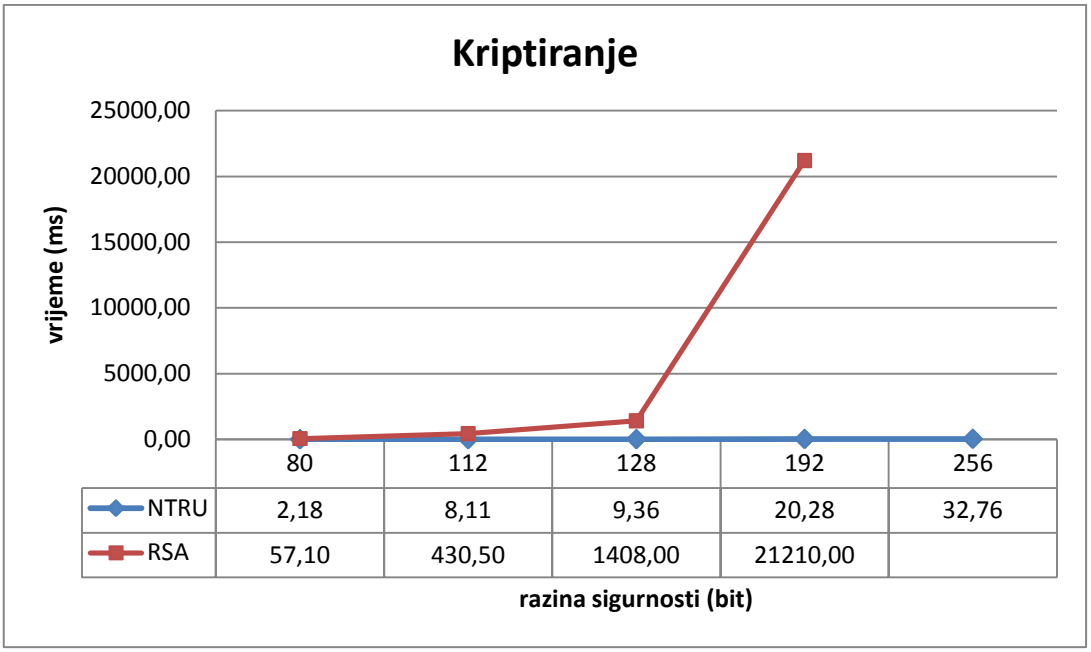
	RSA[16]	**NTRU** [22]
Generiranje ključeva	$O(n^4)$	$O(N^3)$
Kriptiranje	$O(n^2)$	$O(N^2)$
Dekriptiranje	$O(n^3)$	$O(N^2)$
Veličina ključa	n	$N \log_2 p + N \log_2 q$
Ekspanzija poruke	1	$\log_p q$

Na slikama 11.7, 11.8 i 11.9 grafički je prikazana usporedba brzine RSA i NTRU kriptosustava. S obzirom da NTRU odjednom može kriptirati manje količine podataka od RSA, za RSA se čisti tekst uzimao tako da odgovara veličini teksta za NTRU kriptosustav.

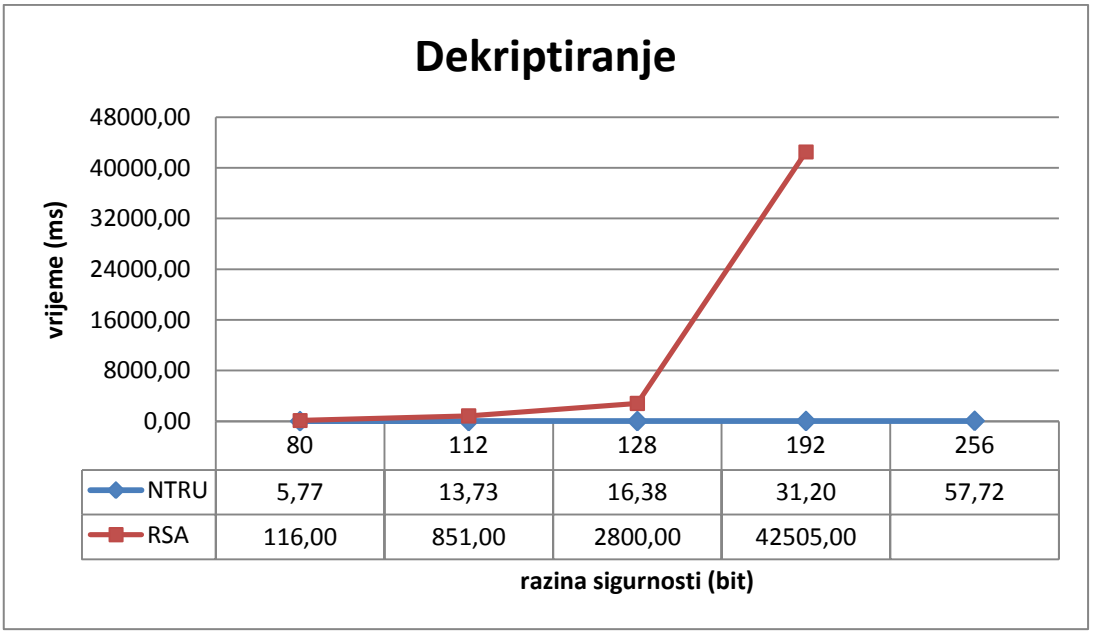


Slika 11.7 Usporedba brzine generiranja ključeva za NTRU i RSA

[16] Preuzeto sa RSA Laboratories: http://www.rsa.com/rsalabs/node.asp?id=2215



Slika 11.8 Usporedba brzine kriptiranja za NTRU i RSA kriptosustav



Slika 11.9 Usporedba brzine dekriptiranja za NTRU i RSA kriptosustav

Kao što se iz priloženih grafova može vidjeti, NTRU je puno brži od RSA kriptosustava. Za razinu sigurnosti od 80 bitova NTRU je 11 puta brži u generiranju ključeva, 26 puta brži prilikom kriptiranja i 20 puta brži u dekriptiranju od RSA kriptosustava. Za razinu sigurnosti od 192 bita, razlika je još veća: generiranje

ključeva je 827 puta brže za NTRU kriptosustav, kriptiranje je 1045 puta brže za NTRU, a dekriptiranje je čak 1362 puta brže kod NTRU kriptosustava!

12. Zaključak

Danas u eri Interneta, ne može se zamisliti život bez korištenja asimetrične kriptografije. Pojavom kvantnih računala danas korišteni asimetrični kriptosustavi više neće biti sigurni. Za NTRU kriptosustav se vjeruje da će biti siguran i u vrijeme kvantnih računala, a zbog velike efikasnosti i malog zauzeća memorije već se počeo upotrebljavati u zaštiti bežičnih mreža. Implementacijom kriptosustava pokazalo se da je NTRU kriptosustav puno efikasniji od RSA. Za razinu sigurnosti od 80 bitova, NTRU je 20ak puta brži od RSA, a za razinu sigurnosti od 192 puta NTRU je skoro 1000 puta brži od RSA kriptosustava! NTRU će se zasigurno naći u širokoj primjeni u skoroj budućnosti.

13. Literatura

[1] Shor, P. W. „Algorithms for quantum computation: discrete logarithm and factoring“. Proc. of the 35th Annual Symposium on Foundations of Computer Science, 1994., str. 124-134. Dostupno na Internet adresi [29.5.2010.]: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.47.3862

[2] NIST Physics Laboratory: „NIST demonstrates universal programmable quantum processor for quantum computers“, studeni 2009. Dostupno na Internet adresi [11.6.2010.]: http://www.nist.gov/physlab/div847/quantumprocessor_111609.cfm

[3] Wikipedia the free encyclopedia: „Cryptography“. Dostupno na Internet adresi [29.5.2010.]: http://en.wikipedia.org/wiki/Cryptography

[4] Budin, L. „Operacijski sustavi 2“, predavanja. Zagreb, Fakultet elektrotehnike i računarstva, 2003.

[5] Škorić, N. „Kriptoanaliza kroz primjere“. Seminarski rad. Zagreb, Fakultet elektrotehnike i računarstva, 2005. Dostupno na Internet adresi [29.5.2010.]: http://os2.zemris.fer.hr/algoritmi/simetricni/2005_skoric/seminar/index.html

[6] Krivačić, V. „Kriptiranje eliptičkim krivuljama“. Diplomski rad. Zagreb, Fakultet elektrotehnike i računarstva, 2006. Dostupno na Internet adresi [29.5.2010.]: http://os2.zemris.fer.hr/algoritmi/asimetricni/2006_krivacic/Diplomski_Krivacic.htm

[7] Wikipedia the free encyclopedia: „Cryptographic primitive“. Dostupno na Internet adresi [29.5.2010.]: http://en.wikipedia.org/wiki/Cryptographic_primitive

[8] Dujella, A. „Diskretna matematika“, predavanja. Dostupno na Internet adresi [29.5.2010.]: http://web.math.hr/~duje/diskretna/diskretna.pdf

[9] Krešić-Jurić, S. „Algebarske strukture“, predavanja. Split, Prirodoslovno-matematički fakultet, 2009. Dostupno na Internet adresi [29.5.2010.]: http://www.pmfst.hr/~skresic/Algebra/Algebarske_strukture.pdf

[10] Žubrinić, D. „Linearna algebra“, predavanja. Zagreb, Fakultet elektrotehnike i računarstva, 2002.

[11] Micciancio, D.; Goldwasser, S. „Complexity of lattice problems: a cryptographic perspective“. Springer, 2002.

[12] Pancheva, K. „On lattices, codes and Regev's cryptosystem“. Bachelor thesis. University of Technology Darmstadt, 2007. Dostupno na Internet

adresi [30.5.2010.]: https://www.cdc.informatik.tu-darmstadt.de/reports/reports/Krasimira_Pancheva.bachelor.pdf

[13] Hoffstein, J.; Pipher, J.; Silverman, J. H. „An introduction to mathematical cryptography“. Springer, 2008.

[14] Wikipedia the free encyclopedia: „Trapdoor function“. Dostupno na Internet adresi [31.5.2010.]: http://en.wikipedia.org/wiki/Trapdoor_function

[15] Lloyd, S. „Quantum Information Science“. MIT Lecture notes, 2009. Dostupno na Internet adresi [31.5.2010.]: http://web.mit.edu/2.111/www/notes09/spring.pdf

[16] Wikipedia the free encyclopedia: „Knapsack problem“. Dostupno na Internet adresi [31.5.2010.]: http://en.wikipedia.org/wiki/Knapsack_problem

[17] Regev, O. „LLL algorithm“. Lattices in computer science, lecture 2. Tel Aviv University, 2004. Dostupno na Internet adresi [31.5.2010.]: http://www.cs.tau.ac.il/~odedr/teaching/lattices_fall_2004/ln/lll.pdf

[18] Schnorr, C. P.; Euchner, M. „Lattice basis reduction: improved practical algorithms and solving subset sum problems“. Proc. of the 8th International FOCS, 1991., str. 68-85. Dostupno na Internet adresi [31.5.2010.]: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.57.3331

[19] Babai, L. „On Lovasz' lattice reduction and the nearest lattice point problem“. Combinatorica, 6(1), 1986., str. 1-13. Dostupno na Internet adresi [31.5.2010.]: http://www.csie.nuk.edu.tw/~cychen/Lattices/On%20lovasz%20lattice%20reduction%20and%20the%20nearest%20lattice%20point%20problem.pdf

[20] Hrg, D. „Simulator kvantnog računala“. Diplomski rad. Zagreb, Fakultet elektrotehnike i računarstva, 2004. Dostupno na Internet adresi [31.5.2010.]: http://os2.zemris.fer.hr/kvant/2004_hrg/diplomski.pdf

[21] Jakuš, M. „Kvantna kriptografija“. Seminarski rad. Zagreb, Fakultet elektrotehnike i računarstva, 2004. Dostupno na Internet adresi [31.5.2010.]: http://os2.zemris.fer.hr/kvant/2004_jakus/index.html

[22] Hoffstein, J.; Pipher, J.; Silverman, J. H. „NTRU: A ring-based public key cryptosystem“. Proc. of the 3rd International Symposium on Algorithmic Number Theory, 1998., str 267-288. Dostupno na Internet adresi [31.5.2010.]: http://www.securityinnovation.com/cryptolab/pdf/ANTS97.pdf

[23] NTRU Cryptosystems, Inc. „The NTRU public key cryptosystem – a tutorial“. Dostupno na Internet adresi [31.5.2010.]: http://www.securityinnovation.com/pdf/Ntru_Public_Key_Cryptosystem_Tutorial.pdf

[24] I. Malović. „Asimetrični kriptosustavi zasnovani na rešetkama“. Seminarski rad. Fakultet elektrotehnike i računarstva, 2009.

[25] Proos, J. A. „Imperfect decryption anda n attack on the NTRU encryption sheme“. IACR (International association for Cryptologic Research) Cryptology ePrint Archive, 2003. Dostupno na Internet adresi [5.6.2010.]: http://eprint.iacr.org/2003/002.pdf

[26] Mersin, A. „The comparative performance analysis of lattice based NTRU cryptosystem with other asymmetrical cryptosystems“. Master thesis. Izmir Institute of Technology, 2007. Dostupno na Internet adresi [1.6.2010.]: http://library.iyte.edu.tr/tezler/master/bilgisayaryazilimi/T000609.pdf

[27] Whyte, W; et al. „IEEE P1363.1 Draft 10: Draft Standard for Public Key Cryptographic Techniques Based on Hard Problems over Lattices“. 2008. Dostupno na Internet adresi [1.6.2010.]: http://eprint.iacr.org/2008/361.pdf

[28] Howgrave-Graham, N.; Silverman, J. H.; Whyte, W. „A Meet-in-the-middle attack on an NTRU private key“ NTRU Cryptosystems Technical Report #004, Version 2, 2003. Dostupno na Internet adresi [1.6.2010.]: http://securityinnovation.com/cryptolab/pdf/NTRUTech004v2.pdf

[29] Monteverde, M. „NTRU software implementation for constrained devices“. Master thesis. Katholieke Universiteit Leuven, Belgija, 2008. Dostupno na Internet adresi [5.6.2010.]: http://www.cosic.esat.kuleuven.be/publications/thesis-161.pdf

[30] Howgrave-Graham, N. „A hybrid lattice-reduction and meet-in-the-middle attack against NTRU“. CRYPTO 2007., str. 150-169. Dostupno na Internet adresi [5.6.2010.]: http://grouper.ieee.org/groups/1363/lattPK/submissions/crypto2007.pdf

[31] Howgrave-Graham, N.; Silverman, J. H.; Singer, A; Whyte, W. „NAEP: Provable security in the presence of decryption failures“. IACR (International association for Cryptologic Research) Cryptology ePrint Archive, 2003. Dostupno na Internet adresi [5.6.2010.]: http://eprint.iacr.org/2003/172.pdf

[32] Howgrave-Graham, N.; Silverman, J. H.; Whyte, W. „Choosing parameter sets for NTRUEncrypt with NAEP and SVES-3“. CT-RSA, 2005. Dostupno na Internet adresi [5.6.2010.]: http://eprint.iacr.org/2005/045.pdf

[33] Hoffstein, J.; Pipher, J.; Silverman, J. H. „NSS: The NTRU Signature Sheme“. Proc. Of Eurocrypt '01, LNCS 2045. Dostupno na Internet adresi [5.6.2010.]: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.25.9263

[34] Gentry, C.; Jonsson, J.; Stern, J.; Szydlo, M. „Cryptanalysis oft he NTRU signature Sheme (NSS) from Eurocrypt '01“. Dostupno na Internet adresi [5.6.2010.]: http://www.szydlo.com/nss-break.pdf

[35] Hoffstein, J.; Howgrave-Graham, N.; Pipher, J.; Silverman, J. H.; Whyte, W. „NTRUSign: Digital signatures using the NTRU lattice“. CT-RSA, 2002.

Dostupno na Internet adresi [5.6.2010.]:
http://www.math.brown.edu/~jpipher/NTRUSign_RSA.pdf

[36] Nguyen, P. Q.; Regev, O. „Learning a parallelepiped: Cryptanalysis of GGH and NTRU signatures“. Journal of Cryptology, Volume 22 (2), 2008. Dostupno na Internet adresi [5.6.2010.]:
http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.147.292

[37] Silverman, J. H. „Almost inverses and fast NTRU key creation“. NTRU Cryptosystems Technical Report #014, Version 1, 1999. Dostupno na Internet adresi [5.6.2010.]:
http://www.securityinnovation.com/cryptolab/pdf/NTRUTech014.pdf

Dodatak A – Popis oznaka i kratica

3DES	*Triple DES*	trostruki DES
AES	***A**dvanced **E**ncryption **S**tandard*	napredni standard kriptiranja, simetrični algoritam kriptiranja
DES	***D**ata **E**ncryption **S**tandard*	standard kriptiranja podataka (bivši); simetrični algoritam kriptiranja
DSS	***D**igital **S**ignature **S**tandard*	standard digitalnog potpisa
ECC	***E**liptic **C**urve **C**ryptography*	kriptografija korištenjem eliptičkih krivulja
IDEA	***I**nternational **D**ata **E**ncryption **A**lgorithm*	simetrični kriptosustav
IEEE	***I**nstitute of **E**lectrical end **E**lectronics **E**ngineers*	Institut inženjera elektrotehnike i elektronike
IEEE P1363	Standardization project for public-key cryptography	IEEE odjel za standarde u kriptografiji javnog ključa
LUC	*Lucas Cryptosystem*	kriptosustav zasnovan na Lucasovim nizovima
NAEP	***N**tru **A**symetric **E**ncryption **P**adding*	shema kriptiranja za NTRU propisana IEEE standardom
SVES	***S**hort **V**ector **E**ncryption **S**heme*	instanca NAEP enkripcijske sheme
NIST	*The **N**ational **I**nstitute of **S**tandards and **T**echnologies*	Nacionalni institut standarda i tehnologija (SAD)
NTRU	***N**-th degree **tru**ncated polynomial ring*	asimetrični kriptosustav opisan u ovom radu
nzd		najveći zajednički djelitelj
qubit	***qu**antum **bit***	kvantni bit
RSA	***R**ivest **S**hamir **A**dleman*	asimetrični kriptosustav nazvan prema svojim tvorcima

Dodatak B – Primjeri datoteka

Prilikom spremanja podataka u datoteke rade se pretvorbe opisane u poglavlju 10.5.

U nastavku je prikazan primjer zapisa privatnog i javnog ključa. Svi podaci zapisani su u heksadecimalnoj bazi. N je duljina polinoma, Q i P su veliki i mali modul, a F i H su privatni i javni ključ respektivno.

```
---BEGIN NTRU CRYPTO DATA---
Description:
    Private key

Method:
    NTRU

N:
    a7

Q:
    80

P:
    03

F:
    8988654599444422a91248506a01216a928509998990868296992926248 5
    14162859a645a407
---END NTRU CRYPTO DATA---
```

Slika B.1 Prikaz privatnog ključa u datoteci

```
---BEGIN NTRU CRYPTO DATA---
Description:
    Public key

Method:
    NTRU

N:
    a7

Q:
    80

H:
    4d495e3c4c66420225064a1873720826064605114f2c3e377876401b3468
    4b5d5e3a2b62426323511740606e1d1e320f576520582d352213447a0a7a
    0c305521146c28775529545d465c69334b454f201e54181b5c216f006e77
    0a1e303f451b551120367b793a502702252f38685a445b6d6948270d511b
    190373051401757302161 81b2859071e4643444f295d407f391b2f627642
    16194a182c1125411a540876394a6f0025
---END NTRU CRYPTO DATA---
```

Slika B.2 Prikaz javnog ključa u datoteci

```
---BEGIN NTRU CRYPTO DATA---
Description:
    Crypted file

Method:
    NTRU

Data:
    BhseL3NnEwtKSA8+fEAzD20NXHpQGEpcEyYLfAYNeD1VV1AjHwBJdm87XCdC
    WRMGVWInTRswOn1lOy1wQRIJf0o6ASYpQz43eVgzHkAqS11XBWwreFk0HV9a
    YRQda15XbRUWe2hCciZnQnN1UTJQEj1/JUFiCh1VQHwRVi8bOBAOFGUUbCFf
    V28FYmJVLUoOMhEIUCQNMjF7DDUfVmxIGVA1CRhKegE=

---END NTRU CRYPTO DATA---
```

Slika B.3 Prikaz kriptiranog teksta u datoteci

Dodatak C – Izvorni kod jezgre NTRU-a

```
public class ClassicNtru : INtruEncrypt
{
    private NtruParameters parameters = null;

    public ClassicNtru()
    {
        parameters = null;
    }

    public void Encrypt(TruncatedPolynomial input, out TruncatedPolynomial
        output, PublicKey publicKey)
    {
        output = Encryption(input, publicKey);
    }

    public TruncatedPolynomial Encrypt(TruncatedPolynomial input, PublicKey
        publicKey)
    {
        return Encryption(input, publicKey);
    }

    public void Decrypt(TruncatedPolynomial input, out TruncatedPolynomial
        output, PrivateKey privateKey)
    {
        output = Decryption(input, privateKey);
    }

    public TruncatedPolynomial Decrypt(TruncatedPolynomial input, PrivateKey
        privateKey)
    {
        return Decryption(input, privateKey);
    }

    public void Decrypt(TruncatedPolynomial input, out TruncatedPolynomial
        output, PrivateKey privateKey, TruncatedPolynomial fq)
    {
        output = Decryption(input, privateKey, fq);
    }

    public TruncatedPolynomial Decrypt(TruncatedPolynomial input, PrivateKey
        privateKey, TruncatedPolynomial fq)
    {
        return Decryption(input, privateKey, fq);
    }

    public void GenerateKeys(out PublicKey publicKey, out PrivateKey
        privateKey)
    {
        if (this.parameters == null)
            throw new ArgumentNullException("Parametri kriptosustava nisu
              definirani!");

        if (parameters.dF == null || parameters.dG == null)
            throw new ArgumentNullException("Parametri dF i dG nisu
              definirani!");
```

```
        TruncatedPolynomial f, fp, fq, g;

        if (this.parameters.P == 3)
        {
            int eksponent = Convert.ToInt32(Math.Log(parameters.Q, 2));
            if (Math.Pow(2, eksponent) != Convert.ToDouble(parameters.Q))
                throw new ArgumentException("Parametar q nije potencija broja
                    2!");
            while (true)
            {
                f = PolynomialGenerator.GeneratePolynomial((int)parameters.dF,
                    (int)parameters.dF - 1, parameters.N);
                if (PolynomialOperator.InvMod3(f, out fp))
                {
                    if (PolynomialOperator.InvMod2naR(f, eksponent, out fq))
                        break;
                }
            }
            g = PolynomialGenerator.GeneratePolynomial((int)parameters.dG,
               (int)parameters.dG, parameters.N);
            publicKey = new PublicKey(PolynomialOperator.NormalMultModQ(g, (fq
               * parameters.P), parameters.Q));
            privateKey = new PrivateKey(f, fp);
        }
        else
        {
            while (true)
            {
                f = PolynomialGenerator.GeneratePolynomial((int)parameters.dF,
                    0, parameters.N);
                if (PolynomialOperator.InvMod2(f, out fp))
                {
                    if (PolynomialOperator.InvModP(f, parameters.Q, out fq))
                        break;
                }
            }
            g = PolynomialGenerator.GeneratePolynomial((int)parameters.dG, 0,
               parameters.N);
            publicKey = new PublicKey(PolynomialOperator.NormalMultModP(g, (fq
               * parameters.P), parameters.Q));
            privateKey = new PrivateKey(f, fp);
        }
    }

    public void ImportParameters(Parameter parameters)
    {
        if (parameters is NtruParameters)
            this.parameters = new NtruParameters(parameters);
        else
            throw new ArgumentException("Parametri nisu standardni!");
    }

    public Parameter ExportParameters()
    {
        return this.parameters as Parameter;
    }
```

```
private TruncatedPolynomial Encryption(TruncatedPolynomial input,
    PublicKey publicKey)
{
    if (this.parameters == null)
        throw new ArgumentNullException("Parametri kriptosustava nisu
          definirani!");
    if (this.parameters.P == 2)
    {
        return (PolynomialGenerator.GeneratePolynomial(parameters.dR, 0,
          parameters.N) * publicKey.publicKey +
          input).ModPositiv(parameters.Q);
    }
    else if (parameters.P == 3)
    {
        return (PolynomialGenerator.GeneratePolynomial(parameters.dR,
          parameters.dR, parameters.N) * publicKey.publicKey +
          input).ModPositiv(parameters.Q);
    }
    else
        throw new ArgumentException("Parametar p je pogrešan! P može biti
          samo 2 ili 3");
}

private TruncatedPolynomial Decryption(TruncatedPolynomial input,
    PrivateKey privateKey)
{
    if (this.parameters == null)
        throw new ArgumentNullException("Parametri kriptosustava nisu
          definirani!");
    TruncatedPolynomial output;
    if (this.parameters.P == 2)
    {
        output = PolynomialOperator.NormalMultModPCentered(privateKey.f,
          input, parameters.Q);
        output = Center(output, privateKey);
        output = PolynomialOperator.NormalMultModQCentered(privateKey.fp,
          output, parameters.P);
        return output;
    }
    else if (parameters.P == 3)
    {
        output = PolynomialOperator.NormalMultModQCentered(privateKey.f,
          input, parameters.Q);
        if (privateKey.fp != null)
            output = PolynomialOperator.NormalMultModPCentered
                (privateKey.fp, output, ((NtruParameters)parameters).P);
        else
            output = output.ModNegPos(parameters.P);
        return output;
    }
    else
        throw new ArgumentException("Parametar p je pogrešan! P može biti
          samo 2 ili 3");
}
```

```
    /// <summary>
    /// Centriranje polinoma a
    /// I=fq(1)*[a(1)-p(1)*r(1)*g(1)] (mod q)
    /// Avg = [p(1)*r(1)*g(1)+I*f(1)] / N
    /// Polinom se centrira u interval [Avg - q/2, Avg + q/2]
    /// </summary>
    /// <param name="a"></param>
    /// <param name="privateKey"></param>
    /// <returns></returns>
    private TruncatedPolynomial Center(TruncatedPolynomial a, PrivateKey
        privateKey)
    {
        if ((parameters.dG == null) || (parameters.dF == null))
            throw new Exception("Nisu definirani df i dg!");
        int I, A, f;
        TruncatedPolynomial fq;
        if (!PolynomialOperator.InvModP(privateKey.f, parameters.Q, out fq))
            throw new Exception("Nema Fq!");
        f = 0; A = 0;
        for (int i = 0; i < fq.N; i++)
        {
            f += fq.GetValue(i);
            A += a.GetValue(i);
        }
        I = (f * (A - parameters.P * parameters.dR * (int)parameters.dG)) %
               parameters.Q;
        double Avg1 = (double)(parameters.P * parameters.dR *
               (int)parameters.dG + I * (int)parameters.dF);
        Avg1 = Avg1 / (double)parameters.N;
        double Avg2 = Avg1 + (double)parameters.Q / 2;
        Avg1 = Avg1 - (double)parameters.Q / 2;
        int min, max;
        min = Convert.ToInt32(Math.Ceiling(Avg1));
        max = Convert.ToInt32(Math.Ceiling(Avg2));
        if ((max - min) != parameters.Q)
            throw new Exception("Centriranje nije dalo interval duljine Q
               brojeva");
        TruncatedPolynomial result = new TruncatedPolynomial(a);
        for (int i = 0; i < a.N; i++)
        {
            if (result.GetValue(i) < min)
                result.SetValue(result.GetValue(i) + parameters.Q, i);
            else if (result.GetValue(i) > max)
                result.SetValue(result.GetValue(i) - parameters.Q, i);
        }
        return result;
    }

}
```