Asimetrični kriptografski algoritmi za razmjenu ključeva otporni na napade kvantnim računalom
Tehnička dokumentacija
Verzija 1.0

Studentski tim: Bernard Crnković
Fran Čutura
Maria Krajči
Dominik Matić

Nastavnik: prof. dr. sc. Marin Golub

1. Uvod

1.1. Kriptografija javnog ključa

Kriptografija javnog ključa ili asimetrična kriptografija jedan je od najbitnijih sigurnosnih mehanizama današnjeg Interneta. Temelji se na korištenju para ključeva: javnog ključa koji može biti poznat svima, i tajnog ključa koji je je sakriven od javnosti i poznat samo njegovom vlasniku. Ključevi se u većini slučajeva koriste za razmjenu simetričnog ključa koji se zatim koristi za kriptiranje svih ostalih poruka u komunikaciji.

Pretpostavimo da imamo dvije osobe, Anu i Branka, koji žele komunicirati preko interneta bez da im itko može presresti i pročitati poruke. Kako bi zaštitili poruke od takve vrste prisluškivanja, potrebno ih je kriptirati, no prije samog šifriranja potrebno je razmijeniti simetričan ključ kojim će to i napraviti. Ta se razmjena postiže pomoću asimetričnih ključeva na sljedeći način:

1. Branko generira **simetričan ključ** K
2. Branko koristi Anin **javni ključ** da kriptira ključ K
3. Branko šalje Ani kriptirani ključ K
4. Ana koristi svoj **tajni ključ** da dekriptira kriptirani ključ K
5. Ana i Branko koriste ključ K da kriptiraju sve ostale poruke u komunikaciji

Ova metoda je sigurna jer javni i tajni ključ su iznimno različiti ali i dalje ovise jedan o drugome. Ono što jedan ključ kriptira drugi može dekriptirati, no bilo koji ključ ne može dekriptirati ono što je on sam kriptirao. Također, znajući jedan od ta dva ključa, skoro je nemoguće u razumnom vremenu izračunati onaj drugi. RKazlog tomu je što se ključevi generiraju jednosmjernim matematičkim funkcijama za koje je nemoguće ili još uvijek nepoznato pronaći reverznu funkciju. Najpoznatiji primjer ovakve generacije jest u RSA algoritmu koji koristi problem faktorizacije iznimno velikih brojeva kako bi se smatrao sigurnim. Klasično računalo, da bi pronašlo faktore nekog broja, nema puno boljeg načina od načina grube sile pa je iz tog razloga dovoljno sporo da se ne smatra ozbiljnom prijetnjom takvom načinu generiranja ključeva.

1.2. Kvantna računala

Na isti način kako su klasična računala transformirala svijet u svijet budućnosti, kvantna računala sada počinju raditi isto. Donose potpuno nove načine računanja koji će pokrenuti eksponencijalan napredak u mnogim područjima primjene. Kvantno računalo rješava mnoge probleme koje klasično računalo u principu neće moći nikada riješiti. No, rješavanje nekih od tih problema može uzrokovati nove probleme. Jedan od tih problema je sigurnost.

Kvantno računalo, sa dovoljno kvantnih bitova, ima sposobnost brzo i efikasno razbiti mnoge od postojećih sustava sigurnosti na internetu. Kvantni algoritmi poput Shorovog algoritma predstavljaju ozbiljan problem za mnoge od sadašnjih načina generiranja asimetričnih ključeva. To je nešto s čim se stručnjaci trenutno sukobljavaju jer je samo pitanje vremena kada će biti

moguće konstruirati kvantno računalo sa dovoljno kvantnih bitova za provedbu takvih algoritama u stvarnom svijetu.

1.3. Ciljevi projekta

Cilj ovoga projekta je predstaviti i opisati četiri algoritma pobjednika i pet alternativnih algoritama koji su prošli u treći krug NIST-ovog natječaja za algoritme javnog ključa otpornih na napade kvantnim računalom. Algoritmi će se predstaviti pomoću interaktivne i edukativne desktop aplikacije, a njihovi opisi su sadržani u ovom dokumentu. Osim aplikacije i tehničke dokumentacije, cilj projekta uključuje i izradu prezentacije i web stranice koja sadrži sve spomenute materijale.

2. Algoritmi

2.1. CRYSTALS-KYBER

2.1.1. Općenito

Kyber je skup algoritama za razmjenu ključeva čija se sigurnost postiže pomoću LWE (Learning with Errors) problema. Rješavanje LWE problema svodi se na rješavanje sustava jednadžbi, no ubačena pogreška pri generaciji čini cijeli problem iznimno teškim, čak i kvantnim računalima. Kyber se zapravo temelji na modularnom LWE problemu, a cijeli problem se može svesti na sljedeću jednadžbu:

$$B = As + e \ mod \ q$$

Problem leži u činjenici da ako poznajemo *A* i *B* iznimno je teško odrediti *s*. U izvedbi algoritma konkatenacija *A* i *B* su javni ključ, *s* je tajni ključ, *e* je ubačena greška, a *q* je neki dovoljno velik prost broj. U pravilu su sve navedene varijable vektori, no Kyber za *A* koristi matricu.

Drugi algoritmi i protokoli poput NewHopea i Frodoa koriste prstenastu LWE odnosno običnu LWE varijantu svaka od kojih ima svoje prednosti i mane. Prednosti prstenastog LWE-a su velika učinkovitost na području brzine i veličini ključeva i šifriranog teksta, a mane predstavljaju problem da povećavanjem algebarske strukture algoritma, cijeli mehanizam može postati ranjiviji na napade te problemi skalabilnosti između učinkovitosti i sigurnosti. Obična LWE varijanta ima prednost minimalne strukture koja omogućava jednostavnu skalabilnost, no uz cijenu smanjivanja cjelokupne učinkovitosti. Modularni LWE kojeg Kyber koristi se nalazi negdje između te dvije varijante; struktura je smanjena u odnosu na prstenasti LWE, ali je zato skalabilnost puno bolja, a performanse vrlo slične.

Kyber je također tzv. IND-CCA2 siguran. Značenje toga ćemo objasniti na primjeru.

1. Pretpostavimo da napadač ima sposobnost kriptiranja i dekriptiranja proizvoljnih poruka, s tim porukama smije raditi koje god operacije želi u polinomnom vremenu
2. Također pretpostavimo da imamo i žrtvu koja koristi IND-CCA2 siguran algoritam za kriptiranje poruka, te je njime generirao javni i tajni ključ
3. Napadač zatim žrtvi pošalje dvije proizvoljne nekriptirane poruke
4. Žrtva nasumično bira jednu od tih poruka i kriptira ju svojim tajnim ključem
5. Napadač sada opet smije kriptirati i dekriptirati proizvoljne poruke (osim poruka koje je poslao žrtvi) i izvršavati proizvoljne operacije
6. Napadač pogađa koju je poruku žrtva izabrala u koraku 4

Ukoliko napadač u prosjeku više puta netočno odgovori nego točno, algoritam je IND-CCA2 siguran. Bitno je spomenuti da su CCA sigurni algoritmi takozvano *aktivno sigurni*. Aktivna sigurnost podrazumijeva otpornost ne samo na pasivne napadače koji prisluškuju, nego i na aktivne koji se ubacuju u kanal komunikacije i pokušavaju mijenjati poruke. IND-CCA2 sigurnost osigurava da bilo koja promjena šifriranog teksta **ne** vodi predvidljivoj promjeni dekriptiranog teksta.

2.1.2. Svojstva

Pošto algoritam u mnogim svojim koracima koristi iznimno brzu varijantu diskretne Fourierove transformacije, ocjena performansi algoritma većinski ovisi o performansama korištenih kriptografskih primitiva. Svi kriptografski primitivi koje Kyber koristi dolaze iz Keccak obitelji algoritama. Iz tog razloga, algoritam je iznimno brz na računalima koja imaju sklopovsku podršku za izvedbe takvih kriptografskih funkcija.

2.1.3. Sigurnost

Kyber ima tri varijante, svaka od kojih predstavlja tri različite razine sigurnosti, a one su redom po jačini Kyber512, Kyber768 i Kyber1024. Kyber se može 'pobjediti' na dva načina: pronalazeći i iskorištavajući greške ili mane u temeljnim kriptografskim primitivima ili rješavajući modularni LWE problem. Oba slučaja su teška za izvesti.

	core-SVP (klasično)	**core-SVP (kvantno)**	**Razina sigurnosti**
Kyber512	111	100	1 (AES-128)
Kyber768	181	164	3 (AES-192)
Kyber1024	254	230	5 (AES-256)

2.1.4. Prednosti i mane

Teško je Kyberu pronaći manu, ali zato ima mnogo prednosti. Zbog korištenja brze Fourierove transformacije i brzih Keccak algoritama, performanse su mu izvrsne pri svakoj razini implementacije. Također, ne smijemo zaboraviti na IND-CCA2 svojstvo koje donosi još jednu dodatnu razinu sigurnosti. Uz sve to, algoritam je i vrlo skalabilan gdje povećanje razine sigurnosti zahtjeva samo promjenu dimenzija matrice.

2.2. FrodoKEM

2.2.1. Općenito

FrodoKEM je Microsoftova obitelj mehanizama za generiranje i razmjenu ključeva. U središtu mehanizma nalazi se FrodoPKE, shema kriptiranja javnim ključem temeljena na dobro poznatom LWE problemu. Za razliku od Kybera, koji se temelji na modularnom LWE, FrodoKEM se temelji na LWE problemu nad nekonstrukturiranim rešetkama. Takav nedostatak algebarske strukture rezultira jednostavnošću algoritama što smanjuje potencijalnu ranjivost cijelog mehanizma. Još jedna posljedica jednostavnosti je iznimno efikasna skalabilnost, gdje je za povećanje sigurnosne razine potrebno promijeniti samo nekoliko parametara. FrodoKEM algoritmi su također i IND-CCA2 sigurni.

2.2.2. Svojstva

Nedostatak algebarske strukture rešetke znači da algoritmi ne mogu koristiti brzu Fourierovu transformaciju za generiranje glavne matrice. Zbog toga njena generacija postaje glavni čimbenik u evaluaciji performansi algoritama. Matrica se generira pomoću kriptografskih primitivnih funkcija na dva moguća načina: AES i SHAKE.

Na procesorima koji imaju sklopovsku potporu za brže izvođenje kriptografskih operacija, AES inačica je značajno brža. No, na procesorima bez sklopovske potpore, SHAKE je u pravilu brži. U svakom slučaju, vrijeme izvođenja algoritama je dovoljno brzo da ostane kompetentno sa ostalim algoritmima.

2.2.3. Sigurnost

FrodoKEM nudi tri varijante za tri razine sigurnosti, to su redom: FrodoKEM-640, FrodoKEM-976 i FrodoKEM-1344 koji odgovaraju razinama sigurnosti AES-128, AES-192 odnosno AES-256. Kao i za Kyber, jedine moguće slabe točke algoritma leže u rješavanju temeljnog LWE problema, te u eksploataciji kriptografskih primitiva.

2.2.4. Prednosti i mane

Najveća prednost FrodoKEM-a leži u njegovoj jednostavnosti; čini ga naročito skalabilnim te eliminira potencijalne ranjivosti koje proizlaze iz algebarske strukture. No, bitno je istaknuti da nedostatak strukture rezultira nešto duljim vremenom izvođenja te većom veličinom ključeva. Tablica ispod uspoređuje veličine ključeva Kybera i FrodoKEM-a (u byteovima).

Shema	secret key	public key	ciphertext
Kyber512	1632	800	768
FrodoKEM-640	19888	9616	9720
Kyber768	2400	1184	1088
FrodoKEM-976	31296	15632	15744
Kyber1024	3168	1568	1568
FrodoKEM-1344	43088	21520	21632

Imajući to na umu, FrodoKEM tim tvrdi da su ključevi i dalje dovoljno mali da ne utječu osjetno na propusnost u uobičajenoj komunikaciji.

2.3. SABER

2.3.1. Općenito

Kriptografija zasnovana na algebarskim rešetkama jedna je od najperspektivnijih kriptografskih obitelji za koje se smatra da pružaju otpor kvantnim računalima. Saber je obitelj kriptografskih algoritama koji se oslanjaju na Module Learning With Rounding problem (Mod-LWR). SABER je jedan od pobjednika drugog kruga kandidata NIST-ovog natječaja za izbor novih asimetričnih algoritama za uspostavu tajnog ključa otpornih na napade kvantnim računalom.

Tim koji je razvio SABER čine članovi koji su bili dio istraživačke skupine COSIC na KU Leuven, Belgija: Jan-Pieter D'Anvers, Angshuman Karmakar, Sujoy Sinha Roy i Frederik Vercauteren. Kasnije su se timu pridružili: Jose Maria Bermudo Mera, Michiel Van Beirendonck i Andrea Basso.

Prvo je nastao Saber.PKE, IND-CPA siguran način šifriranja ključem koji se kasnije pretvara u Saber.KEM, IND-CCA način sigurne enkapsulacije ključem, koristeći Fujisaki-Okamoto transformacije.

Ciljevi SABER-a bili su jednostavnost, djelotvornost i fleksibilnost. Zbog toga svi cjelobrojni moduli su potencije od 2, upotrebom LWR prepolavlja se potrebna količina generiranja slučajnih varijabli u usporedbi s LWE shemama i smanjuje se propusnost te struktura modula pruža fleksibilnost ponovnom upotrebom jedne temeljne komponente za višestruku sigurnost razinama.

SABER nudi tri razine sigurnosti: LightSABER - postkvantna razina sigurnosti slična AES-128, SABER - postkvantna razina sigurnosti slična AES-192, FireSABER: postkvantna razina sigurnosti slična AES-256.

2.3.2. Parametri

Stupanj n = 256 polinomskog prstena Zq [X] = (Xn + 1) i rang l od modula koji određuje dimenziju osnovnog problema rešetke kao l*n. Povećanje dimenzije problema rešetke povećava sigurnost, ali smanjuje točnost.

Moduli koji su uključeni u shemu odabrani su kao potencijali od 2, posebno q = 2 q, p = 2 p i T = 2 T s q> p> T, pa imamo T j p j q. Veći izbor za parametre p i T rezultirat će manjom sigurnošću, ali većom točnošću. Skriptom u pythonu se izračunavaju optimalne vrijednosti za p i T.

Koeficijenti tajnih vektora s i s' određuju se prema centriranoj binomnoj raspodjeli s parametrom μ, gdje je μ < p. Veća vrijednost za μ rezultirat će većom sigurnošću, ali manjom točnošću.

Funkcije za izračunavanje sažetka koje se koriste u protokolu su F, G i H. Funkcije F i H su implementirane pomoću SHA3-256 (s određenom duljinom ulaza), dok je G implementirana pomoću SHA3-512.

Funkcija izlaza gen s mogućnošću proširivanja koja se koristi u protokolu za generiranje pseudoslučajne matrice A od $seed_A$. Implemetirana je pomoću SHAKE-128.

2.3.3. Šifriranje javnim ključem

Saber.PKE (Saber Public Key Encryption) je način šifriranja javnim ključem koji se sastoji od 3 algoritma: Saber.PKE.KeyGen, Saber.PKE.Enc i Saber.PKE.Dec. Saber.PKE.KeyGen generira javni ključ, Saber.PKE.Enc šifrira javnim ključem pri čemu može koristiti zadani agrument r te Saber.PKE.Dec dekriptira javnim ključem.

2.3.4. Enkapsulacija ključem

Saber.KEM (Saber Key-Encapsulation Mechanism) je način enkapsulacije ključem koji se sastoji od 3 algoritma: Saber.KEM.KeyGen, Saber.KEM.Enc i Saber.KEM.Dec. Saber.KEM.KeyGen generira ključ, Saber.KEM.Enc enkapsulira ključem pri čemu koristi Sabre.PKE.Enc te Saber.KEM.Dec dekriptira ključem pri čemu koristi Sabre.PKE.Dec.

2.3.5 Veličina parametara

Saber.PKE:

kategorija sigurnosti	**vjerojatnost neuspjeha**	**klasično računalo core-SVP**	**kvantno računalo core-SVP**	**javni ključ (B)**	**kriptirani ključ (B)**	**dekriptirani ključ (B)**
LightSaber-PKE: l = 2, n = 256, q = 2^{13}, p = 2^{10}, T = 2^3, μ = 10						
1	2^{-120}	2^{118}	2^{107}	672	832 (256)	736
Saber-PKE: l = 3, n = 256, q = 2^{13}, p = 2^{10}, T = 2^4, μ = 8						
3	2^{-136}	2^{189}	2^{172}	992	1248 (288)	1088
FireSaber-PKE: l = 4, n = 256, q = 2^{13}, p = 2^{10}, T = 2^6, μ = 6						
5	2^{-165}	2^{260}	2^{236}	1312	1664 (384)	1472

Saber.KEM:

kategorija sigurnosti	**vjerojatnost neuspjeha**	**klasično računalo core-SVP**	**kvantno računalo core-SVP**	**javni ključ (B)**	**kriptirani ključ (B)**	**dekriptirani ključ (B)**
LightSaber-KEM: l = 2, n = 256, q = 2^{13}, p = 2^{10}, T = 2^3, μ = 10						
1	2^{-120}	2^{118}	2^{107}	672	1568 (992)	736
Saber-KEM: l = 3, n = 256, q = 2^{13}, p = 2^{10}, T = 2^4, μ = 8						
3	2^{-136}	2^{189}	2^{172}	992	2304 (1344)	1088
FireSaber-KEM: l = 4, n = 256, q = 2^{13}, p = 2^{10}, T = 2^6, μ = 6						
5	2^{-165}	2^{260}	2^{236}	1312	3040 (1760)	1472

2.3.6. Očekivana razina sigurnosti

Izračunavanje sažetka javnog ključa u vektor K osigurava da ključ ovisi o unosu obje strane i nudi višestruku zaštitu.

Najviše osigurava da napadač nije u mogućnosti koristiti unaprijed izračunate slabe vrijednosti na više ciljeva prilikom traženja grešaka u dešifriranju.

Budući da se izvršava u stalnom vremenu dobra je obrana protiv napada kojim se koriste podaci o vremenu određenih izračuna kako bi se dobili podaci o dugotrajnom tajnom ključu.

Još jedna obrana od napada je to što Saber ne koristi nijedan kod za ispravljanje pogrešaka, on izbjegava sve napade povezane s maskiranjem bloka za ispravljanje pogrešaka.

Napadi koji se rade množenjem polinoma ili matrice s tajnim ključem otežani su na način da slučajnim redoslijedom izvode operacija ili uvođenjem lažne operacije, koje se mogu primijeniti na maskirane implementacije.

2.4. SIKE

2.4.1. Općenito

SIKE je algoritam za enkapsulaciju ključem zasnovan na svojstvima izogenija supersingularnih eliptičkih krivulja, od kojih je najpoznatiji protokol za razmjenu ključeva SIDH (supersingular isogeny Diffie-Hellman key exchange) kojeg su 2011. godine predložili Luca de Feo, David Jao i Jerome Plut.

Inačica SIKE (Supersingular Isogeny Key Encapsulation) se nalazi među alternativnim kandidatima u trećoj rundi NIST-ovog natječaja za izbor novih asimetričnih algoritama za uspostavu tajnog ključa otpornih na napade kvantnim računalom.

Tim koji je razvio SIKE čine: David Jao, Reza Azarderakhsh, Matthew Campagna, Craig Costello, Luca De Feo, Basil Hess, Aaron Hutchinson, Amir Jalali, Koray Karabina, Brian Koziel, Brian LaMacchia, Patrick Longa, Michael Naehrig, Geovandro Pereira, Joost Renes, Vladimir Soukharev i David Urbanik.

SIKE sadrži dva algoritma: CPA-zaštićeni algoritam šifriranja javnim ključem SIKE.PKE i CCA-siguran mehanizam za enkapsulaciju ključevima SIKE.KEM, svaki instanciran s četiri skupa parametara: SIKEp434, SIKEp503, SIKEp610 i SIKEp751.

2.4.2. Parametri

Javni parametri u SIKE-u su: dvije pozitivne cijele vrijednosti e_2 i e_3 koje definiraju konačno polje F_{p2} gdje je $p = 2^{e2}3^{e3} - 1$, početna supersingularna eliptička krivulja $E_0 = F_{p2}$, skup od tri x koordinate koje odgovaraju točkama u E_0 $[2^{e2}]$ te skup od tri x koordinate koje odgovaraju točkama u E_0 $[3^{e3}]$.

2.4.3 Tajni ključ

Algoritmi PKE i KEM zahtijevaju dva tajna ključa, sk_2 i sk_3, koji se koriste za izračunavanje 2^{e2}-izogenije i 3^{e3}-izogenije.

Supersingualni izogenijski par ključeva sastoji se od tajnog ključa sk_I, koji je cijeli broj i skupa od tri x koordinate $pk_I = (x_P, x_Q, x_R)$.

2.4.4. Šifriranje javnim ključem

Algoritam 1 definira shemu šifriranja javnim ključem PKE = (Gen, Enc, Dec). Veličina prostora za poruke $M = \{0, 1\}^n$ i funkcija F koja preslikava zajedničku tajnu *j* u nizove bitova ostaju neodređeni, a konkretni izbori odgovaraju implementaciji. Funkcija Enc generira slučajne vrijednosti u sk_2. U slučaju mehanizma enkapsulacije ključem želimo te slučajne vrijednosti proslijediti kao ulaz pa se tada koristi write $(c_0, c_1) \leftarrow \text{Enc}(pk_3, m, sk_2)$.

Algorithm 1: PKE = (Gen, Enc, Dec)

```
function Gen
    Input: ()
    Output: (pk3, sk3)
1   sk3 ←R K3
2   pk3 ← isogen3(sk3)
3   return (pk3, sk3)

function Enc
    Input: pk3, m ∈ M, r ∈ K2
    Output: (c0, c1)
4   sk2 ← r
5   c0 ← isogen2(sk2)
6   j ← isoex2(pk3, sk2)
7   h ← F(j)
8   c1 ← h ⊕ m
9   return (c0, c1)

function Dec
    Input: sk3, (c0, c1)
    Output: m
10  j ← isoex3(c0, sk3)
11  h ← F(j)
12  m ← h ⊕ c1
13  return m
```

2.4.5. Enkapsulacija ključem

Algoritam 2 definira ključni mehanizam enkapsulacije KEM = (KeyGen, Encaps, Decaps) primjenom transformacija Hofheinza, Hövelmannsa i Kiltza u PKE. Ova transformacija je modificirana uključivanjem pk_3 u ulaz u funkciju G i pojednostavljivanjem "ponovnog šifriranja". Veličina M = {0, 1}^{n} i funkcije G i H ostaju neodređene, a konkretni izbori odgovaraju implementaciji.

Algorithm 2: KEM = (KeyGen, Encaps, Decaps)

```
function KeyGen
    Input: ()
    Output: (s, sk3, pk3)
1   sk3 ←R K3
2   pk3 ← isogen3(sk3)
3   s ←R {0,1}^n
4   return (s, sk3, pk3)

function Encaps
    Input: pk3
    Output: (c, K)
5   m ←R {0,1}^n
6   r ← G(m ‖ pk3)
7   (c0, c1) ← Enc(pk3, m; r)
8   K ← H(m ‖ (c0, c1))
9   return ((c0, c1), K)

function Decaps
    Input: (s, sk3, pk3), (c0, c1)
    Output: K
10  m' ← Dec(sk3, (c0, c1))
11  r' ← G(m' ‖ pk3)
12  c'0 ← isogen2(r')
13  if c'0 = c0 then
14      K ← H(m' ‖ (c0, c1))
15  else
16      K ← H(s ‖ (c0, c1))
17  return K
```

2.4.6. Veličina parametara

Osam skupova parametara su SIKEp434, SIKEp434_compressed, SIKEp503, SIKEp503_compressed, SIKEp610, SIKEp610_compressed, SIKEp751 i SIKEp751_compressed, nazvani tako zbog duljina bitova osnovnog polja.

2.4.7. Očekivana razina sigurnosti

Pod pretpostavkom da dostupna memorija dopušta pohranu 280 jedinica, može se zaključiti da SIKEp434 i SIKEp610 udovoljavaju odgovarajućim sigurnosnim zahtjevima NIST-ovih kategorija 2 i 4. Nedavni članak Costella, Longe, Naehriga, Renesa i Virdije tvrdi da SIKEp751, za koji je isprva predloženo da zadovoljava razinu 3, zapravo ispunjava zahtjeve NIST-ove kategorije 5.

Početni prijedlog SIKE koristio je asimptotsku složenost Tanijevog kvantnog algoritma zajedno sa čvrstim donjim granicama broja kvantnih vrata korištenih u Fp-množenju i broju takvih množenja u tipičnom izračunu izogenije. Nedavno istraživanje Jaquesa i Schancka provodi mnogo detaljniju analizu najpoznatijih kvantnih algoritama za računsko rješavanje supersingualnog problema izogenije. Jaques i Schanck predlažu model kvantnog računanja koji dopušta izravnu usporedbu između kvantnih i klasičnih algoritama. Relevantno ograničenje za podudaranje sigurnosnih kategorija NIST nameće ograničenje dubine kvantnih krugova između 264 i 296. Dopuštajući dubine 296, Jaques i Schanck zaključuju da niti jedan poznati kvantni algoritam ne može slomiti SIKE u njihovom modelu proračuna s manje od 2143 klasičnih vrata i 2207 klasičnih vrata za SIKEp434 i SIKEp610. Stoga su ova dva skupa parametara prikladna za NIST kategorije 1 i 3. Skripte za izradu istih tablica za SIKEp503 i SIKEp751 sugeriraju da kvantni algoritam ne može slomiti one s manje od 2146, odnosno 2272 klasičnih vrata, što potvrđuje da su ovi skupovi parametara prikladni za NIST kategorije 2 i 5.

Analiza bočnih kanala napada cilja razne fizičke pojave koje emitira kriptografska implementacija kako bi se otkrile kritične interne informacije o uređaju. Informacije o potrošnji energije, informacije o vremenu i elektromagnetsko zračenje se emitiraju dok se izvode kriptografski algoritmi. Općenito, kriptografija zasnovana na izogeniji svodi se na dva izračuna: generiranje tajne jezgre i izračunavanje izogenije velikog stupnja nad tom jezgrom. U shemama poput SIKE pronađena je tajna jezgra izračunavanjem množenja dvostrukih točaka preko torzijske osnove. Dakle, postoje 2 opća pristupa koje napadač može iskoristiti za napad na sigurnost kriptosustava analizom bočnog kanala: otkrivanje dijelova skrivene točke jezgre i otkrivanje tajnih izogenih šetnji tijekom izračuna izogenije. Ciljajući ove dijelove SIKE kriptosustava, napadač bez sumnje ima pristup širokom spektru snaga, vremenu, kvaru i raznim drugim bočnim kanalima. Konstantne implementacije pomoću konstantnog skupa pokazale su se dobrom protumjerom protiv SPA i vremenskih napada. Međutim, napade diferencijalne analize snage i napade ubrizgavanja kvara puno je teže obraniti.

2.5. HQC

2.5.1. Općenito

HQC (Hamming Quasi-Cyclic) je algoritam šifriranja javnim ključem dizajniran da pruži sigurnost od napada klasičnih i kvantnih računala. HQC je jedan od zamjenskih kandidata za pobjednike trećeg kruga kandidata NIST-ov natječaja za izbor novih asimetričnih algoritama za uspostavu tajnog ključa otpornih na napade kvantnim računalom.

Tim koji je razvio HQC čine članovi: Carlos Aguilar Melchor, Nicolas Aragon, Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Jurjen Bos, Jean-Christophe Deneuville, Arnaud Dion, Philippe Gaborit, Jérôme Lacan, Edoardo Persichetti, Jean-Marc Robert, Pascal Véron i Gilles Zémor.

HQC započinje s verzijom PKE koja se kasnije razvija u KEM-DEM koji postiže IND-CCA2 razinu sigurnosti.

Glavne prednosti HQC-a u odnosu na postojeće kriptosustave koji se temelje na kodu su njegovo smanjenje IND-CPA na jednostavniji teoretski problem kodiranja, uspješan protiv napada koji cilja oporavak skrivene strukture koda, mala veličina javnog ključa, točna procjena stope neuspjeha u dekripciji i djelotvorne implementacije temeljene na klasičnim algoritmima dekodiranja.

Ograničenje HQC algoritma (barem za verziju PKE) je niska enkripcijska stopa. Moguće je šifrirati 256 bitova otvorenog teksta prema zahtjevu NIST-a, ali povećavajući tu stopu također se povećavaju parametri.

HQC nudi tri razine sigurnosti: HQC-128, HQC-192, HQC-256.

2.5.2. Parametri

Postoji nekoliko skupova parametara koji ciljaju različite razine sigurnosti s DFR-om povezanim s tim razinama sigurnosti. Predloženi skupovi parametara pokrivaju sigurnosne kategorije 1, 3 i 5 (za 128, 192 i 256 bitova sigurnosti). Za svaki skup parametara, parametri su odabrani tako da minimalni radni faktor najpoznatijeg napada premaši sigurnosni parametar. Smatra se $w = O\sqrt{n}$.

U tablici veličina parametara, n_1 označava duljinu Reed-Salomonovog koda, n_2 duljinu Reed-Mullerovog koda tako da duljina spojenog koda C iznosi n_1n_2 (ambijentalni prostor ima duljinu n, a najmanji primitivni prosti broj veći od n_1n_2 je potreban kako bi se izbjegli algebarski napadi). w je težina n-dimenzionalnih vektora x, y, w_r težina r_1 i r_2 te je prema tome $w_e = w(e)$ za ovaj kriptosustav. p_{fail} je vjerojatnost pogreške prilikom šifriranja i dekripcije javnim ključem.

2.5.3. Veličina parametara

instanca	n_1	n_2	n	w	$w_r = w_e$	razina sigurnosti	p_{fail}
hqc-128	46	384	17.669	66	75	128	$< 2^{-128}$

hqc-192	56	640	35.851	100	114	192	$< 2^{-192}$
hqc-256	90	640	57.637	131	149	256	$< 2^{-256}$

instanca	veličina pk (bit)	veličina sk (bit)	veličina ct (bit)	veličina ss (bit)
hqc-128	2.249	40	4.481	64
hqc-192	4.522	40	9.026	64
hqc-256	7.245	40	14.469	64

2.5.4. Očekivana razina sigurnosti

Praktična složenost problema SD-a za Hammingovu metriku široko je proučavana više od 50 godina. Najučinkovitiji napadi temelje se na dekodiranju skupa informacija, a prvo je tehniku uveo Prange 1962., kasnije je poboljšao Stern, zatim Dumer. Nedavni radovi sugeriraju složenost reda $2^{cw\,(1\,+\,negl\,(1))}$, za neku konstantu c. Posebni dio koji se fokusira na režim w = negl (n) potvrđuje ovu formulu, uz jaku ovisnost između c i brzine k/n koda koji se koristi.

Kvazi-ciklički kodovi imaju posebnu strukturu koja može potencijalno otvaraju vrata za specifične strukturne napade. Prvi generički napad je DOOM napad koji zbog cikličnosti podrazumijeva dobitak od O (√n) (kada je dobitak u O (n) za MDPC kodove jer se kod generira na osnovi vektora male težine). Također je moguće razmotriti napade na oblik polinoma koji generira cikličku strukturu. Takvi napadi posebno su učinkoviti kada je polinom x^n - 1 ima mnogo faktora niskog stupnja. Ti napadi postaju nedjelotvorni čim x^n – 1 ima samo dva nesvodljiva faktora oblika (x - 1) i x^n-1 + x^n-2 + ... + x + 1, što je slučaj kada je n prost i q generira multiplikativnu skupinu (Z /nZ). Takvi su brojevi poznati do vrlo velikih vrijednosti.

HQC ima različite skupove parametara koji pružaju 128 (kategorija 1), 192 (kategorija 3) i 256 (kategorija 5) bitova klasične (tj. pretkvantne) sigurnosti. Kvantna sigurnost dobiva se dijeljenjem sigurnosnih bitova za dva (uzimajući drugi korijen složenosti). Kada je w = O (√n), najpoznatiji napadi imaju složenost u $2^{-t\,ln\,(1\,-\,R)\,(1\,+\,o\,(1))}$ gdje t = O (w), a R je stopa koda. U ovom algoritmu je t = 2w i R = 1/2 za redukciju na problem 2-DQCSD, a t = 3wr i R = 1/3 za 3-DQCSD problem. Uzimajući u obzir DOOM napad, kao i činjenicu da se smatraju uravnoteženim vektorima (x, y) i (r_1, e, r_2) za napad (koji zahtijeva samo vrlo mali faktor jer slučajne riječi imaju dobru vjerojatnost uravnoteženja na svakom bloku), ova složenost se mora podijeliti s približno √n (do faktora polinoma). Varijabla o(1) je log $(()^2 / ())$ i log $(()^3 / ())$ za 2-DQCSD i 3-DQCSD probleme. Dakle, smanjenje sigurnosti čvrsto odgovara generičkim primjerima klasičnih problema 2-DQCSD i 3-DQCSD.

2.6. Classic McEliece

2.6.1. Općenito o algoritmu

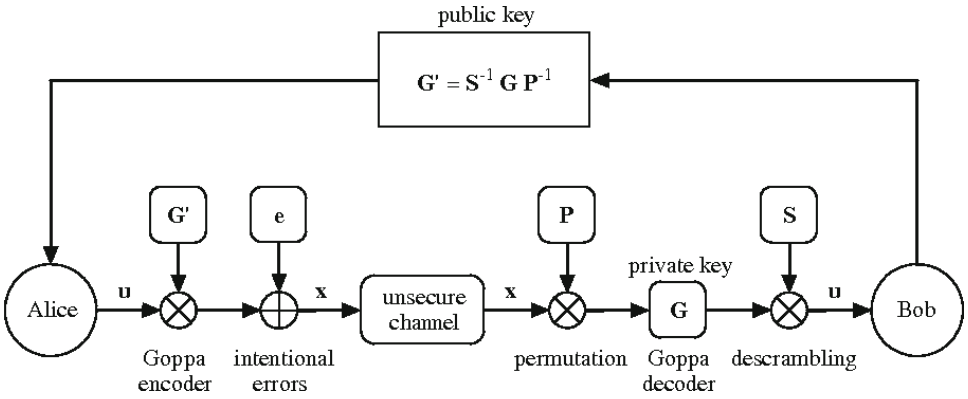
Classic McEliece je familija kandidata asimetričnih kriptografskih algoritama. Ideju samog algoritma osmislio je Robert McEliece 1978. godine, no tada nije ozbiljno razmotren njegov rad sve do danas kada nam se postepeno približava doba kvantnih računala. Unatoč njegovoj starosti, ovaj algoritam je nadmašio više od 40 godina napredaka u matematici i brojne pokušaje kriptoanalize.

2.6.2. Parametri

Algoritam je baziran na Goppinom kodiranju. To je vrsta zaštitnog kodiranja koju je definirao sovjetski matematičar Valerii Denisovich Goppa. Radi se o cikličkom kodiranju baziranom na (nadopuni) (dakle, generirajuća matrica se dobiva rotacijom bitova posebnog polinoma g(x)). Goppin kod je definiran s dva elementa:

1. Polinomom stupnja t nad Galoisovim poljem GP (2^m) bez višestrukih nultočaka
2. Nizom L od n jedinstvenih elemenata iz tog Galoisovog polja koji nisu korijeni izabranog polinoma.

Komunikacijski kanal možemo vizualizirati sljedećom slikom:



Tako definirani kod ima distancu 2t + 1 te može ispraviti ((2t + 1) - 1)/2 pogrešaka u kodnim riječima duljine n - mt te se zbog mogućnosti ispravljanja velikog broja pogrešaka koristi u McEliece kriptosustavu. Ovo je primjer parametara koje koristi prvi od algoritama iz McEliece kriptosustava:

inačica	**m**	**n**	**t**	**f**	**F**
mceliece348864	12	3488	64	$z^{12}+z^3+1$	$y^{64}+y^3+y+z$

Zanimljivo svojstvo McEliece kriptosustava jest mogućnost da dekriptira bilo koji kriptirani tekst sa privatnim ključem neovisno o tome je li on stvarno nastao kriptiranjem odgovarajućim javnim ključem. Upravo se to svojstvo koristi za digitalno potpisivanje u Niederreiter-ovoj varijaciji McEliece kriptosustava.

Također, razina sigurnosti koju nudi je IND-CCA2 što znači da je otporan na sposobnost dedukcije privatnog ključa iz teoretski kooperativnog vlasnika koji za njega dekriptira arbitrarne kriptate (koje napadač može adaptirati prema prethodno poslanim kriptatima po svojoj želji).

2.6.3. Svojstva

Jedno od negativnih svojstava svih post-kvantnih kriptografskih algoritama koje ovdje obrađujemo jest veličina javnog ključa. Doduše, McEliece javni ključevi se kreću u veličinama od 26 KB do 1.35 MB ovisno o odabranim prethodno navedenim parametrima. Točne veličine su vidljive u sljedećoj tablici (vrijednosti veličina su u oktetima):

Inačica	Javni ključ	Privatnog ključ	Kriptat	Ključ sjednice
mceliece348864	261120	6492	128	32
mceliece348864f	261120	6492	128	32
mceliece460896	524160	13608	188	32
mceliece460896f	524160	13608	188	32
mceliece6688128	1044992	13932	240	32
mceliece6688128f	1044992	13932	240	32
mceliece6960119	1047319	13948	226	32
mceliece6960119f	1047319	13948	226	32
mceliece8192128	1357824	14120	240	32
mceliece8192128f	1357824	14120	240	32

Ove veličine ne čine ovaj kriptosustav pogodnim za ugrađene memorijski ograničene uređaje koji žele postići otpornost na napade kvantnih računala. Istovremeno, nije pogodan niti za blockchain tehnologiju gdje je mala veličina ključa od velike važnosti.

2.6.3. Stabilnost

Slabosti ovog kriptosustava obrađene su u brojnim analizama (https://classic.mceliece.org/papers.html). Pronađene su ranjivosti u generaliziranim Reed-Solomon kodnim sustavima, no još do danas nisu pronađene iste za poseban podskup već spomenutih Goppa kodova sa opisanim svojstvima. Kada se koristi kod jačine k = (0.8 + o(1)) * n kada n -> ∞, čak i Prangeov napad (1962., dekodiranje skupa informacija), koji jest prijetnja za sve sustave bazirane na kodovima, ima eksponencijalnu složenost $(5+o(1))^t$ (u najgorem slučaju, dakle je poznato da je njegovo rješavanje NP-težak problem). 't' = (0.2 + O(1)) * n / log2(n) je jednak broju pogrešaka korištenih u sustavu. To je najbolji poznati napad kada struktura kodirajuće matrice nije dobro dizajnirana te su, naravno, uzimajući to u obzir, odabrani McEliece službeni polinomi čiji su nazivi navedeni u tablici iznad. Što se tiče kvantnih računala, ona kao i u drugim shemama reduciraju taj eksponencijalni faktor na pola $(5+o(1))^{t/2}$ no to i dalje ostaje u domeni eksponencijalno složenih problema ako se ne pronađe efikasniji algoritam. Možemo zaključiti da je McEliece baziran na vrlo čvrstim temeljima, odnosno po svojoj prirodi teškom problemu ukoliko se ne pokaže da je milenijski problem P = NP istinit.

2.7. BIKE

2.7.1 Općenito o algoritmu

BIKE je još jedna kodna shema bazirana na kvazi cikličkim polinomima koja nudi tri verzije (BIKE-1, BIKE-2 te BIKE-3). Nastao je 2017. godine kao produkt rada sedamnaestero doktoranada iz Francuske, Sjedinjenih Država, Izraela i Njemačke pri čemu je dio autora sudjelovalo i u dizajniranju i HQC algoritma.

2.7.2 Parametri

Veličina bloka je r za kodne riječi duljine n = 2 * r, a Hammingova težina w = sqrt(n) te vrijedi da je w paran, a w/2 neparan. Težinu unesenog vektora pogreške označavamo sa t, a duljinu dijeljene tajne sa l. M = $\{0, 1\}^l$ je prostor mogućih poruka, a K = $\{0, 1\}^l$ je prostor mogućih ključeva.

Dekoder je, dakle, zasnovan na principu QC-MDPC (kvazi ciklički srednje gusti kôd provjere pariteta) i ispravljanju namjerno unesenih pogrešaka sa zadovoljavajuće visokom vjerojatnošću. Javni je ključ sastavljen od $h_1h_0^{-1}$ te se on šalje sugovorniku u komunikaciji. On zatim koristi njega da stvori poruku kao ($e_0 + e_1 h_1h_0^{-1}$, $h_1h_0^{-1}$) te se oslanjamo na činjenicu da je vrlo težak problem razlučiti kriptat od slučajnog vektora ((e_0, e_1), $h_1h_0^{-1}$). Privatni ključ za dekriptiranje je sastavljen kao konkatenacija dvaju ciklučkih matrica polinoma H_0 i H_1.

2.7.3 Sigurnost

Razlikuje se od prethodno obrađenog HQC i McEliece algoritama po svojstvu da nije IND-CCA2 siguran, već IND-CCA1 siguran. To je i za očekivati s obzirom na to da je skup važećih matrica za njegovo generiranje zapravo nadskup od McEliece algoritma te su zahtjevi na sigurnost time i slabiji. Znači da nije otporan na postepenu dedukciju ključa iz modela u kojem napadač može slati proizvoljan broj kriptiranih poruka. To za posljedicu ima potrebu da se pronalazi novi ključ iz prostora ključeva za svaku sjednicu komunikacije odnosno poruku.

2.7.4 Prednosti i nedostaci

Prethodno navedene tri verzije BIKE-a idejno odgovaraju NIST-ovim razinama sigurnosti: 1, 3 i 5 (ekvivalent AES-128, AES-192, AES-256 razinama sigurnosti). Parametri za petu razinu sigurnosti odabrani su naknadno na NIST-ov zahtjev na kraju drugog kruga selekcije. Ono što BIKE stvarno čini posebnim od ostalih kandidata jest trud uložen u optimizaciju njegovog izvođenja kako bi se nadomjestila nužnost jednokratnih ključeva. Jan Richter-Brockmann i Tim Güneysu su autori VHDL hardverske implementacije BIKE-a visokih performansi čiji je kod dostupan na Githubu.

Veličine BIKE ključeva, iako velike u usporedbi sa tradicionalnim metodama kriptiranja, nisu znatno veće od prosječnog RSA 4096-bitnog ključa. Konkretne veličine elemenata u sustavu možemo vidjeti u tablici ispod izražene u bitovima:

Element	Size	Razina 1	Razina 2	Razina 3
Privatni ključ	l+ w * □$\log_2(r)$□	2,244	3,346	4,640

Javni ključ	r	12,323	24,659	40,973
Kriptat	r + l	12,579	24,915	41,229

2.8. NTRU

2.8.1 Općenito o algoritmu

NTRU je prvi kriptosustav koji nije baziran na faktorizaciji velikih brojeva ili problemu diskretnih logaritama, koristi se kriptografijom baziranom na polinomijalnim prstenima koja se može opisati i rešetkama. Njegova sigurnost se dijelom oslanja na (eksperimentalno promatranoj) činjenic da je za većinu rešetaka iznimno teško pronaći kratke vektore. Za originalan razvoj algoritma, 1996 godine, zaslužni su Jeffrey Hoffstein, Jill Pipher, i Joseph H. Silverman. Ime NTRU daje nam naslutit na koji način kriptosustav radi. NTRU je kratica za eng. **N**-th degree **T**runcated polynomial **R**ing **U**nits. Operacije šifriranja su bazirane nad objektima u kvocijentnom polinomijalnom prstenu. Algoritam se hvali svojom brzinom i djelotvornosti.

2.8.2. Parametri

- N – polinom unutar prstena R ima stupanj N-1 (javno)
- q – veliki modul (javno)
- p – mali modul (javno)
- f – polinom koji čini privatni ključ (privatno)
- g – polinom koji generira javni ključ h iz f (privatno, ali jednokratno)
- h – javni ključ, također polinom (javno)
- r - nasumičan polinom za prikrivanje (privatno, ali jednokratno)
- d – koeficjent

2.8.3. Primjer procedure šifriranja

1) Stvaranje ključeva

Prvi korak: Biraju se dva mala polinoma f i g unutar kvocijentnog prstena polinoma. Odabrani polinomi bi trebali biti tajni i njihov inverz mora biti postojan.

Drugi korak: traženje inverza polinoma f modul q (koji označavamo sa f_q) i inverza f modul p (koje označavamo sa f_p).

Treći korak: računanje javnog ključa

Privatni ključ: par polinoma (f, f_p)
Javni ključ: $h = p * ((f_q * g) \bmod q)$

Primjer:

1. f(x) = x^6−x^4+x^3+x^2−1 (privatno) i g(x) = x^6+x^4−x^2−x.

2. $f_q(x)$ = f(x)^ −1 (mod q) = 8x^6 + 26x^5 + 31x^4 + 21x^3 + 40x^2 + 2x + 37 (mod 41).
$f_p(x)$ = f(x)^ −1 (mod p) = x^6 + 2x^5 + x^3 + x^2 + x + 1 (mod 3) - privatni ključ

3. $h(x) = p * f_q * g \pmod q$ = 20x^6 + 40x^5 + 2x^4 + 38x^3 + 8x^2 + 26x + 30 (mod 41) (javni ključ)

2) Šifriranje

Prvi korak: poruka se pretvara u polinom koji označavamo sa *m*, koeficijenti tog polinoma su koeficijent *modul p* gdje je *p* između *-p/2* i *p/2*

Drugi korak: Nasumično se bira još jedan mali polinom *r*

Treći korak: Šifrirana poruka se računa na sljedeći način: *e = r * h + m (modul q)*

Primjer: m(x) = −x^5 + x^3 + x^2 − x + 1 (Poruka koju želimo poslati)

r(x) = x^6 − x^5 + x − 1 (Nasumičan mali polinom)

e(x) ≡ 31x^6+19x^5+4x^4+2x^3+40x^2+3x+25 (mod 41) (Šifrirana poruka)

3) Dešifriranje

Prvi korak: Koristeći privatni polinom f izračunava se polinom a = f * e (modul q). Koeficijenti polinoma a moraju biti u intervalu –q/2 do q/2 uključivo.

Drugi korak: Izračunava se polinom b = a (modul p)

Treći korak: Orginalna poruka se dobiva na sljedeći način: $c = f_b * b$ (modul p) Primjer:

a ≡ x^6 + 10x^5 + 33x^4 + 40x^3 + 40x^2 + x + 40 (mod 41).

b = a(mod p) = x^6 + 10x^5 − 8x^4 − x^3 − x^2 + x − 1 (mod 3)

$c = f_p(x)*b(x)$ ≡2x^5 + x^3 + x^2 + 2x + 1 (mod 3).

m(x) =−x^5 + x^3 +x^2 − x + 1. pri redukciji modulo 3 koeficjenti su iz intervala [-1,1]

2.8.4. Prednosti i ograničenja

1) Otpornost na napade kvantnim računalima (Trenutno ne postoji poznat napad koristeći kvantna računala na kriptosustav NTRU)

2) Brzina i učinkovito šifriranje i dešifriranje, u softverskim i hardverskim implementacijama

3) Mogućnost korištenja jednokratnih ključeva zbog iznimne brzine stvaranja istih (“jeftino” je stvoriti ključ)

4) NTRU za potrebe šifriranja koristi iznimno malo memorije, njegova implementacija je pogodna za uređaje sa limitiranim resursima poput mobilnih uređaja ili pametnih kartica.

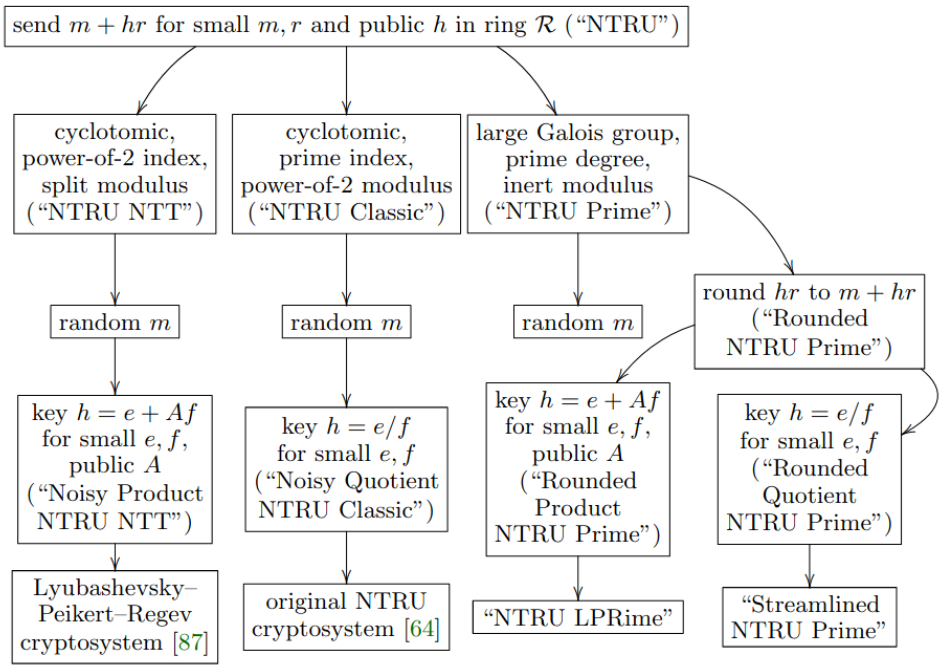
2.9. NTRU Prime

2.9.1 Općenito o algoritmu

Nekoliko kriptosustava zasnovanih na idealnoj rešetci slomljeno je nedavnim napadima koji iskorištavaju posebne strukture prstenova koji se koriste u tim kripto sustavima. Iste strukture također se koriste u vodećim prijedlozima za kriptografiju zasnovanu na post-kvantnoj rešetci, uključujući klasični kriptosistem NTRU. Kriptosustav NTRU Prime ispravlja NTRU tako što ne koristi prstenove sa problematičnim strukturama i predlaže kriptosustav visokog stupnja sigurnosti, na bazi idealne rešetke primarnog stupnja inertnog modula velike Galoisove skupine.

Autori NTRU Prime kriptosustava su Daniel J. Bernstein, Chitchanok Chuengsatiansup, Tanja Lange & Christine van Vredendaal.

Prikaz NTRU obitelji kriptosustava



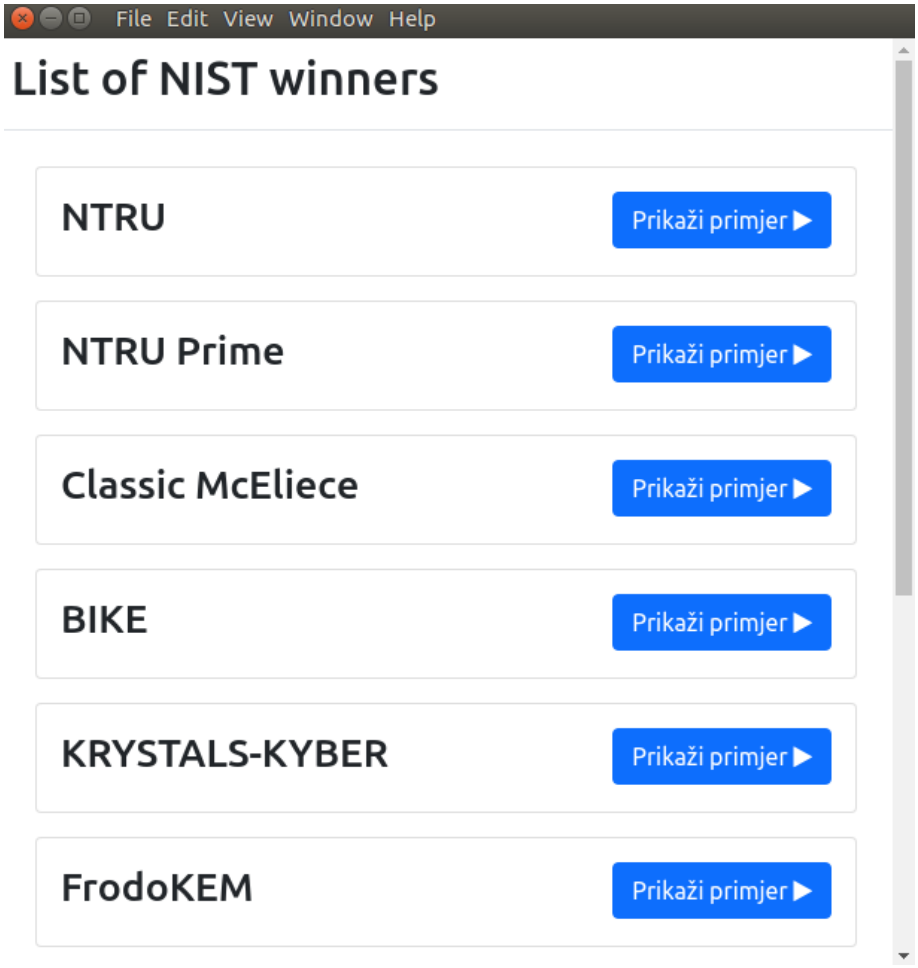
NTRU prime se koristi poljima oblika (Z/q)[x]/($x^p - x - 1$), gdje je p prost broj

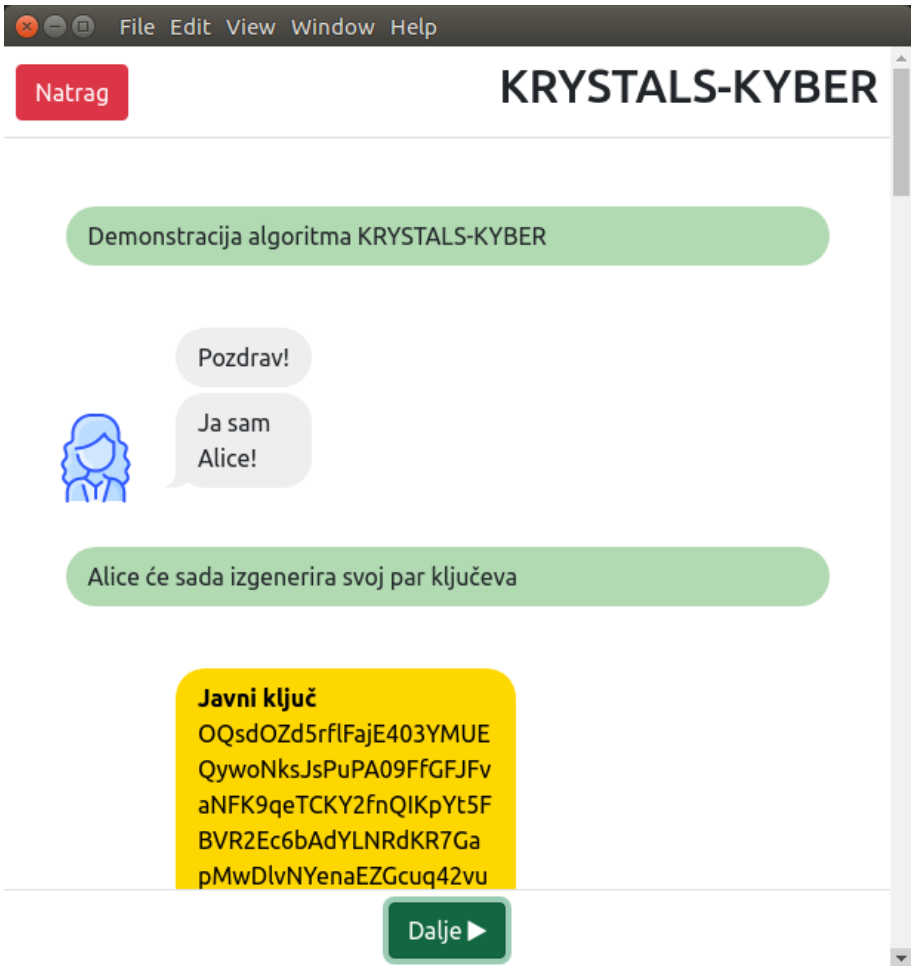
2.9.2 Sigurnost

NTRU Prime obećaje poboljšanu sigurnost za razliku od klasične varijante NTRU-a uz najmanji trošak performansi.

3. Grafičko sučelje

Aplikacija je zamišljena kao simulacija razgovora između Alicea i Boba tijekom kojeg oni razmjenjuju svoje post-kvantne kriptografske ključeve te ih koriste za enkapsulaciju dijeljene tajne koju onda koriste za komunikaciju simetričnom enkripcijom - konkretno u našem slučaju koristimo Rijndaelov blok kod za simetrično kriptiranje komunikacije nakon razmjene ključeva. Početna stranica aplikacije je izbornik koji nudi odabir željenog algoritma. Odabirom jedne opcije, stvara se simulacija razmjene ključeva za taj algoritam.





Također postoji mogućnost samostalnog generiranja, enkapsulacije i dekapsulacije ključeva koristeći po želji odabran algoritam. Također je moguće šifrirati željenu datoteku dijeljenom tajnom pomoću AES algoritma.

4. Implementacija i izazovi pri implementaciji

4.1. Backend

Prvi korak pri realizaciji projekta bio je osposobljavanje različitih knjižnica algoritama. Izvorne kodove algoritama skinuli smo sa njihovih stranica te su neki od njih već nudili opciju stvaranja dijeljene knjižnice. Za one koji to nisu nudili, morali smo to sami napraviti. Proces stvaranja dijeljene knjižnice (za GNU/Linux) opisan je u nastavku:

1. skupiti sve relevantne .c datoteke, u pravilu su to sve koje ne sadržavaju funkciju main
2. prevesti .c datoteke u .o objekte: gcc –fpickamaterina -c [sve .c datoteke]
3. prevesti objekte u dijeljenu knjižnicu: gcc -shared –o libimelibraryja.so [sve .o datoteke]

Kako bi opracijski sustav (GNU/Linux) mogao koristiti knjižnice potrebno je ažurirati LD_LIBRARY_PATH dodavajući novu .conf datoteku u /etc/ld.so.conf.d/ direktorij u kojoj se nalazi putanja direktorija u kojemu se nalaze novostvorene knjižnice. Nakon toga potrebno je izvršiti naredbu *sudo ldconfig* i knjižnica je spremna za uporabu.

Nakon što imamo .so datoteku potrebno je za nju napisati sučelje koje će frontend dio aplikacije moći koristiti. Pošto frontend radimo u Electronu, koji se temelji na node.js-u, za sučeljavanje algoritama koristimo N-API knjižnicu. Drugim riječima, napisali smo C program koji poziva funkcije dijeljenih knjižnica koji smo onda povezali sa ostatkom aplikacije koristeći N-API knjižnicu.

Kako bi N-API znao koristiti naš C program, bilo je potrebno stvoriti binding.gyp datoteku u kojoj navodimo sve parametre prevođenja programa kao i knjižnice koje on koristi. Priličan izazov je bio pronaći funkcionirajuću konfiguraciju binding.gyp datoteke koju možete vidjeti na sljedećoj slici.

```
{
    "targets": [
        {
            "target_name": "main",
            "sources": [ "./src/main.c" ]
        },
        {
            "target_name": "kyber512",
            "sources": [ "./src/algosi/kyber/kyber512.c", "./src/algosi/kyber/randombytes.c", "./src/algosi/kyber/randombytes.h" ],
            "link_settings": {
                "libraries": [ "-lkyber", "-lssl", "-lcrypto", "-lb64" ],
                "ldflags": [ "-L../src/algosi/lib" ]
            }
        },
        {
            "target_name": "kyber768",
            "sources": [ "./src/algosi/kyber/kyber768.c", "./src/algosi/kyber/randombytes.c", "./src/algosi/kyber/randombytes.h" ],
            "link_settings": {
                "libraries": [ "-lkyber", "-lssl", "-lcrypto", "-lb64" ],
                "ldflags": [ "-L../src/algosi/lib" ]
            }       },
        {
            "target_name": "kyber1024",
            "sources": [ "./src/algosi/kyber/kyber1024.c", "./src/algosi/kyber/randombytes.c", "./src/algosi/kyber/randombytes.h" ],
            "link_settings": {
                "libraries": [ "-lkyber", "-lssl", "-lcrypto", "-lb64" ],
                "ldflags": [ "-L../src/algosi/lib" ]
            }
        },
        {
            "target_name": "frodo640",
            "sources": [ "./src/algosi/FrodoKEM/frodo640.c" ],
            "link_settings": {
                "libraries": [ "-lfrodo640", "-lssl", "-lcrypto", "-lb64" ],
                "ldflags": [ "-L../src/algosi/lib" ]
            }
        },
        {
            "target_name": "frodo976",
            "sources": [ "./src/algosi/FrodoKEM/frodo976.c" ],
            "link_settings": {
                "libraries": [ "-lfrodo976", "-lssl", "-lcrypto", "-lb64" ],
                "ldflags": [ "-L../src/algosi/lib" ]
            }
        },

        ...
```

Bitno je spomenuti kako je izuzetno važan redoslijed navođenja atributa "libraries" za svaki pojedini algoritam. Potrebno je to učiniti kao na slici te ukoliko se ne navede tim redoslijedom, sučelje neće funkcionirati.

4.2. Frontend

Za ostvarenje grafičkog sučelja koristili smo JavaScript radni okvir Electron u kombinaciji sa Vue radnim okvirom. Electron radni okvir je projekt baziran na Chromium web pregledniku koji se razvija kao zaseban projekt namijenjen za desktop aplikacije. Programiranje u tom radnom okviru svodi se na programiranje u web browseru specifično za jednu interaktivnu web stranicu. Electron nam omogućava da serviramo JS, HTML i CSS resurse direktno s lokalnog diska na računalu domaćinu samome sebi što pruža korisniku osjećaj nativne aplikacije iako se zapravo koristi preglednikom. Razlika je u tome što Electron omogućava programeru bolju integraciju sa operacijskim sustavom korisnika te lakši pristup lokalnom datotečnom sustavu (što bi u klasičnom pregledniku bio sigurnosni rizik). Zato naša aplikacija omogućava da demonstrativno kriptiramo datoteke na lokalnom korisnikovom računalu.

U pozadini Electron poziva nativni NodeJS API (NAPI) za sučeljavanje nativne dijeljene knjižnice (.so za GNU/linux) koje smo prethodno sagradili.

Za jednostavniju modularnu gradnju komponenata naše interaktivne aplikacije koristili smo moderan frontend framework VueJS.

Projekt prevodimo pomoću modernog alata Webpack koji služi za rezoluciju ovisnosti u JavaScript datotekama te okuplja resurse potrebne za ispravno izvođenje koda. Također minificira kod prilikom prevođenja. Nakon prevođenja koda, koristimo electron-builder alat koji iz našeg projekta gradi AppImage (moderan format za distribuciju aplikacija) kako bi svi resursi i aplikacija bili dostupni u samo jednoj izvršnoj datoteci.

5. Izvori

1. Roberto Avanzi, Joppe Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint,Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, Damien Stehlé, "CRYSTALS-Kyber", *Algorithm Specifications And Supporting Documentation*

2. FrodoKEM, Learning With Errors Key Encapsulation Algorithm, *Specifications And Supporting Documentation*

3. *https://ntru.org/f/ntru-20190330.pdf*
https://link.springer.com/chapter/10.1007%2FBFb0054868
https://asecuritysite.com/encryption/lattice?n=7&p=3&q=41
http://people.scs.carleton.ca/~maheshwa/courses/4109/Seminar11/NTRU_presentation.pdf
http://sigurnost.zemris.fer.hr/algoritmi/asimetricni/2010_malovic/NTRU.pdf
https://eprint.iacr.org/2016/461.pdf

4. Andrea Basso, Jose Maria Bermudo Mera, Jan-Pieter D'Anvers, Angshuman Karmakar, Sujoy Sinha Roy, Michiel Van Beirendonck, Frederik Vercauteren, SABER_KEM (Round 3), *Supporting Documentation*

5. David Jao, Reza Azarderakhsh, Matthew Campagna, Craig Costello, Luca De Feo, Basil Hess, Aaron Hutchinson, Amir Jalali, Koray Karabina, Brian Koziel, Brian LaMacchia, Patrick Longa, Michael Naehrig, Geovandro Pereira, Joost Renes, Vladimir Soukharev, David Urbanik, SIKE, *Supporting Documentation*

6. Carlos Aguilar Melchor, Nicolas Aragon, Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Jurjen Bos, Jean-Christophe Deneuville, Arnaud Dion, Philippe Gaborit, Jérôme Lacan, Edoardo Persichetti, Jean-Marc Robert, Pascal Véron, Gilles Zémor, HQC, *Supporting Documentation*

7. Classic McEliece https://classic.mceliece.org/nist/mceliece-20201010.pdf

8. BIKE https://bikesuite.org/files/v4.1/BIKE_Spec.2020.10.22.1.pdf

