FFI-NOTAT

Eksternnotat 20/02837

Testing av kvantesikre kandidatalgoritmer på en mikrokontroller

– «Norges sikreste chat»

Forfattere
Ella Wolff Kristensen, Ludvig Ellingsen, Martin Strand

19. november 2020

Godkjennere
Raymond Haakseth, *forskningsleder*
Dokumentet er elektronisk godkjent og har derfor ikke håndskreven signatur.

Emneord
Kryptografi, kryptologi, datasikkerhet, mikrokontrollere, testing

Sammendrag
Både sivilt og militært baserer vi oss stadig mer på mange store og små digitale enheter som kommuniserer med hverandre. Enhetene skal helst være billige, og med det følger det begrenset ytelse og lagringsplass. Likevel skal disse enhetene kommunisere sikkert med hverandre, og da behøver vi sikre kryptosystemer. En stor trussel mot mange av dagens kryptosystemer er utviklingen av kvantedatamaskiner. Dersom en tilstrekkelig avansert kvantedatamaskin blir tilgjengelig vil den kunne forsere mange av nåtidens veletablerte kryptosystemer. For å komme utviklingen i forkjøpet har NIST satt i gang en prosess der målet er å utvikle og standardisere ett eller flere kvantesikre kryptosystemer for nøkkelutveksling og signering. I vårt prosjekt har vi tatt i bruk et utvalg av NIST-kandidatene på små mikrokontrollere med begrenset ytelse og lagringsplass. Demonstrasjonen består av en enkel nøkkelutveksling mellom to slike små datamaskiner, og med en påfølgende mulighet til å sende krypterte meldinger mellom dem. Vi har gjort målinger av alle de kandidatene i runde tre av prosessen som kan kjøre på vår maskinvare. Blant finalistene er det algoritmer som har akseptable kjøretider også på høye sikkerhetsnivåer. I det midterste sikkerhetsnivået varierer kjøretidene for kryptering fra 5 millisekunder og helt opp til 1,2 sekunder. Størrelsen på chiffertekstene på samme nivå varierer fra 486 byte til over 15 kB.

Innhold

Forord

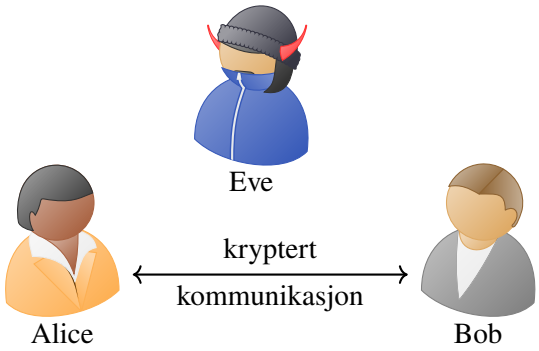
Ella Wolff Kristensen og Ludvig Ellingsen, som har stått for det vesentlige av det tekniske arbeidet bak dette notatet, var sommerstudenter ved FFI sommeren 2020. Som veileder er det en stor glede å få takke studentene Ella og Ludvig for strålende innsats og resultater i løpet av en hektisk og spesiell sommer 2020. Hvert år får FFI hundrevis av sommerjobbsøknader fra de dyktigste studentene i Norge, og det er inspirerende å få jobbe med studenter som tar utfordringer på strak arm, og ikke gir seg når verken andre, tredje eller fjerde innfallsvinkel på et problem fungerer.

Vi må også takke Matthias Kannwischer i PQM4-teamet for entusiastisk støtte og svar på spørsmål. Det samme gjelder vår egen Sondre Engebråten, som har vært til uvurderlig hjelp ved valg og forståelse av maskinvare til dette prosjektet.

Kjeller, september 2020
Martin Strand

1 Introduksjon

Vi ønsker at to parter – la oss kalle dem Alice og Bob – kan kommunisere hemmelig og trygt med hverandre uten at en eventuell tyvlytter Eve er i stand til å forstå innholdet i samtalen. Dette er illustrert i figur 1.1.



Figur 1.1 Sikker kommunikasjonskanal.

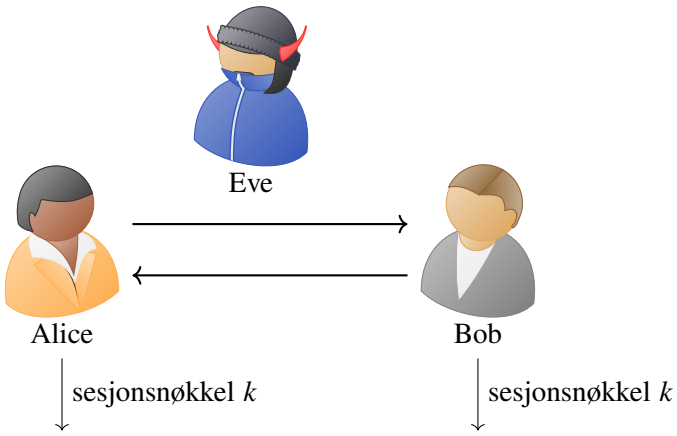
Kimen til svært mange interessante problemer og momenter ligger begravd i den tilsynelatende enkle åpningen over. For det første er det enkle eksempelet utgangspunktet for all sikker kommunikasjon, og dermed også forutsetningen for all samhandling i møte med en motivert motstander.

Videre må vi si noe om hvilke evner vi tillegger tyvlytteren. Det er vanlig å si at hun kontrollerer kommunikasjonskanalen, og derfor kan lese og modifisere alt av nettverkstrafikk. Vår utfordring er dermed å gjøre det slik at Eve ikke kan hente ut noe meningsfylt informasjon fra trafikken hun leser, og at vi kan oppdage det dersom hun forsøker å endre noe. I tillegg legger vi her til grunn at hun har tilgang til en effektiv kvantedatamaskin. Vi kommer tilbake til hva det innebærer. Til sist tar vi høyde for at Alice og Bob kanskje kan være små droner eller sensorer, og derfor ikke har store beregningsressurser. Når vi nå har introdusert disse begrensningene for oss selv, er vi klare til å skissere hvordan vi ønsker å løse problemet, og hvordan vi med denne rapporten bringer dette praktiske problemet for det offisielle Norge litt nærmere en løsning.

Det er vanlig å benytte et symmetrisk kryptosystem for å sende data i kommunikasjonskanalen. I et symmetrisk kryptosystem bruker begge parter den samme nøkkelen for å kryptere og dekryptere hverandres meldinger. Vi kaller denne nøkkelen for en sesjonsnøkkel. For å oppnå en slik sikker kommunikasjonskanal må Alice og Bob ha avtalt sesjonsnøkkelen på forhånd. Noen alternativer for å avtale nøkkelen er å møte hverandre fysisk, eller å sende en kurér med nøkkelen. Det vil være tidkrevende, og må følge strenge rutiner for å unngå at nøkkelen kan bli kompromittert. Til samme formål på Internett gjør man nøkkelutvekslingen digitalt, og da brukes asymmetrisk kryptografi. Det innebærer at partene ikke trenger å starte med en delt hemmelighet.

Selve nøkkelutvekslingen skal foregå sikkert uten at en eventuell lytter skal forstå hva sesjonsnøkkelen er. Figur 1.2 illustrerer nøkkelutveksling på en forenklet måte. Hva slags informasjon som blir delt mellom Alice og Bob avhenger av hvilken metode for nøkkelutveksling de bruker. Diffie-Hellman og RSA er to asymmetriske kryptosystemer som kan brukes til nøkkelutveksling i dag.

Sikkerheten i nøkkeletableringen er altså helt avgjørende for at Alice og Bob kan ha en hemmelig kommunikasjonskanal. Dersom sikkerheten i nøkkelutvekslingen svikter vil lytteren kunne få tilgang



Figur 1.2 Nøkkelutveksling.

til sesjonsnøkkelen, og dermed kunne dekryptere hemmelige meldinger som sendes mellom Alice og Bob i deres kommunikasjonskanal.

En alvorlig trussel mot dagens metoder for nøkkelutveksling er kvantedatamaskiner. Dersom en avansert nok kvantedatamaskin blir tilgjengelig vil den kunne forsere mange av nåtidens asymmetriske kryptosystemer, deriblant Diffie-Hellman og RSA. Vi kommer nærmere inn på dette i avsnitt 2.1. Foreløpig finnes det ikke så kraftige kvantedatamaskiner, men i et føre-var-perspektiv er det nødvendig å anta at det kan være en realitet innen noen tiår. Da må samfunnet være utrustet med kvantesikre kryptosystemer.

Nye, kvantesikre kryptosystemer for nøkkelutveksling er under utvikling og blir standardisert av det amerikanske National Institute of Standards and Technology (NIST). Disse må kunne anvendes der det er nødvendig. Systemene baserer seg på helt andre matematiske prinsipper enn dagens kryptosystemer, og de er lite utprøvde. Derfor er det ikke trivielt at de nye systemene enkelt kan implementeres og brukes på alle digitale enheter, og i hvert fall ikke på alle de små mikrokontrollene som samfunnet er utstyrt med i dag. Dette er motivasjonen bak vårt prosjekt, der vi forsøker å kjøre kandidatalgoritmer fra biblioteket PQM4 på en ARM Cortex M4-prosessor på en Teensy 3.6 mikrokontroller. Vi bruker biblioteket til å bli enig om en hemmelig nøkkel, og fordi vi deretter kan sende meldinger mellom to datamaskiner gjennom mikrokontrollerne har vi noe tabloid døpt demonstrasjonen vår «Norges sikreste chat».

I neste kapittel gir vi ytterligere bakgrunn på kryptografi, kandidatalgoritmene og maskinvaren vi bruker. Kapittel 3 inneholder detaljer på demonstrasjonsprotokollen vår, maskinvaren og hvilke eksterne biblioteket vi benytter. Deretter gir vi tids- og minnemålinger i kapittel 4 før vi konkluderer i kapittel 5.

I denne teksten bruker vi noen sentrale ord slik: En algoritme er en konkret følge av instruksjoner som kjøres fra start til slutt. Vi omtaler så en samling av kryptografiske algoritmer som et kryptosystem (*scheme*). System brukes dermed i en abstrakt mening, i kontrast til en fysisk ting og infrastrukturen rundt en slik eller slike.

2 Bakgrunn

I denne rapporten forholder vi oss til kryptografi for å opprette sikre kanaler mellom to parter. Det vil si at trafikken er uforståelig for andre enn partene, og at de kan oppdage det dersom de mottar uautoriserte meldinger gjennom kanalen. Slik kryptografi kan deles i to store kategorier: symmetrisk og asymmetrisk. Symmetrisk kryptografi kan brukes når partene allerede deler en hemmelighet, og er normalt svært effektiv. Dersom partene imidlertid ikke ennå har noen delt hemmelighet må vi bruke asymmetrisk kryptografi for å etablere en nøkkel. Dette kapittelet gir en oversikt over de vanligste teknikkene i dag, trusselen fra kvantedatamaskiner, og den brede prosessen administrert av NIST for å standardisere nye, kvantesikre algoritmer. Til sist kommer vi litt inn på maskinvaren vi har brukt i eksperimentet.

2.1 Klassisk kryptografi og kvantetrusselen

Med klassisk kryptografi mener vi algoritmer som ble designet for å være sikre mot en klassisk datamaskin, i kontrast til en kvantedatamaskin. De fleste av dagens metoder for asymmetrisk kryptografi er basert på et lite utvalg av svært vanskelige matematiske problemer, blant annet primtallsfaktorisering av høye tall, og diskrete logaritmer. Et gjennombrudd i hvordan klassiske datamaskiner kan løse slike problemer vil føre til at krypterings- og signatursystemet RSA og nøkkelutvekslingsprotokollen Diffie-Hellman – begge avgjørende for blant annet sikkerhet på Internett – kan forseres. I 1994 viste Shor at kraftige kvantedatamaskiner kan løse disse problemene effektivt [24].

En kvantedatamaskin bruker kvantemekanikk til å utføre visse beregninger langt mer effektivt enn en vanlig datamaskin kan. Uten å gå inn i matematiske detaljer gjør det at en kvantedatamaskin kan beregne periodiske sammenhenger i mengder med mye struktur på en vesentlig raskere måte enn vi kan i dag. I dag finnes det bare relativt enkle kvantedatamaskiner, men de utgjør foreløpig ingen trussel: Hittil har de bare greid å gjøre faktorisering som mennesker kunne gjort med hoderegning. Det er vanskelig å si når en tilstrekkelig avansert kvantedatamaskin vil være tilgjengelig, men føre-var-prinsippet dikterer at vi må ta høyde for at det kan skje innenfor den tiden dagens informasjon fortsatt er sensitiv. Slike kvantedatamaskiner kan da knekke flere av de asymmetriske systemene vi bruker i dag.

De asymmetriske kryptosystemene som brukes i dag er altså utfordret av kvantemaskinenes utvikling. Men hva med de symmetriske? Våre etablerte symmetriske systemer er tilsynelatende trygge, selv mot avanserte kvantedatamaskiner. Det vil si at vi kan beholde de symmetriske metodene for å sende og motta kryptert informasjon, for eksempel AES. Kvantedatamaskinene utfordrer AES i noe grad, og en kan ved hjelp av Grovers algoritme [12] angripe AES noe mer effektivt enn tidligere. Dette problemet kan løses ved å bruke AES med 256 bit nøkler. Da mener man at sikkerheten er god nok selv mot en kvantedatamaskin.

2.2 Nye standarder

For å komme utviklingen av kvantedatamaskiner i forkjøpet har NIST satt i gang en prosess for å standardisere et eller flere kvanteresistente, asymmetriske systemer for autentisert nøkkelutveksling. Det ble sendt inn bidrag fra hele verden, og i 2016 ble 69 kandidater kvalifisert til å delta. Tredje runde ble annonsert i juli 2020, med sju finalister og åtte «reservefinalister» [3]. NIST anslår å ha de første standardene klar i løpet av perioden 2022–2024.

Prosessen skal standardisere algoritmer for kryptering og signering. Krypteringsalgoritmene primært er tenkt brukt til å kryptere nøkler for symmetriske algoritmer, og gir derfor bare sikkerhetsgarantier for korte, tilfeldige klartekster. Videre henviser vi til slike som en *Key Encapsulation Mechanism* (KEM). Signatur- og KEM-algoritmene kan brukes sammen for at to aktører kan dele den krypterte sesjonsnøkkelen samtidig som at man er sikre på at motparten er den hun gir seg ut for å være. NIST har til hensikt å standardisere minst én *IND-CCA2*-sikker[1] KEM-protokoll og én *EUF-CMA*-sikker[2] signaturprotokoll [20].

NIST-kandidatene baserer seg på helt andre matematiske problemer enn dagens kryptosystemer. De nye metodene baserer seg på blant annet gitre (*lattices*), multivariate polynomer, feilkorrigerende koder, isogenier på elliptiske kurver og hashfunksjoner. Tabell 2.1 viser finalistene og problemene de er basert på.

KEM		Signatur	
Navn	Problem	Navn	Problem
Classic McEliece [6]	dekodingsproblemet	DILITHIUM [17]	gitter (MLWE, SIS)
KYBER [23]	gitter (MLWE)	FALCON [21]	gitter (NTRU, SIS)
NTRU [29]	gitter (NTRU)	Rainbow [11]	multivar. ligninger (OUV)
SABER [9]	gitter (MLWR)		
BIKE [4]	dekodingsproblemet	GeMSS [8]	multivariate ligninger
FrodoKEM [19]	gitter (LWE)	Picnic [28]	hashfunk., zero-knowledge
HQC [1]	dekodingsproblemet	SPHINCS+ [13]	hashfunksjoner
NTRU Prime [7]	gitter (NTRU)		
SIKE [14]	isogenier		

Tabell 2.1 Finalistene (øverste halvdel) og reservefinalistene (nederste halvdel) i tredje runde av NISTs standardiseringsprosess.

Kandidatene er bedt om å levere ett eller flere parametersett for inntil fem kategorier, som er definert ved hjelp av etablerte algoritmer. For å bli plassert i en kategori må parameterne gjøre det like vanskelig å angripe kandidatsystemet som det ville vært å angripe algoritmen som definerer kategorien, se tabell 2.2.

I tabell 2.1 oppgir vi hvilke matematiske problemer de utvalgte finalistene er basert på. Dette notatet er ikke det riktige stedet å gå i dybden på disse, men vi gir en rask innføring i de ideene og intuisjonen som ligger til grunn. Denne delen forutsetter noe kjennskap til gruppeteori.

[1] IND-CCA2: *Indistinguishability under adaptive chosen ciphertext attack.*
[2] EUF-CMA: *Existensial unforgeability under chosen message attack.*

Sikkerhetsnivå	Tilsvarende styrke
1	AES-128
2	SHA-256
3	AES-196
4	SHA-384
5	AES-256

Tabell 2.2 Definisjonen av sikkerhetsnivåene.

2.2.1 Gitterproblemer

La $n > 0$ være et heltall, og la V være et n-dimensjonalt vektorrom over $\mathbb{R}$ med basis $\{\vec{b}_1, \ldots \vec{b}_n\}$. Et gitter er en undergruppe Λ av V med $\Lambda \simeq \mathbb{Z}^n$, og er gitt ved

$$\Lambda = \left\{ \sum_{i=1}^{n} a_i \vec{b}_i \mid a_i \in \mathbb{Z} \right\}.$$

Det er to grunnleggende beregningsproblemer:

Closest vector problem (CVP) Gitt en vektor $\vec{v} \in V$, finn det gitterpunktet $\vec{\lambda} \in \Lambda$ som ligger nærmest $\vec{v}$. Nærhet måles ofte med den vanlige euklidske metrikken, men kan også bruke andre.

Shortest vector problem (SVP) Finn det korteste gitterpunktet $\vec{\lambda} \neq 0$. Som over kan man formulere problemet med valgfri norm.

Begge disse problemene er vist å være *NP-hard* under rimelige betingelser [2, 26]. Andre problemer som kan reduseres til disse inngår i familien av gitterproblemer.

For NTRU kan man vise at den offentlige nøkkelen genererer en $2N \times 2N$-matrise som representerer et gitter. Dersom man kan finne den korteste vektoren i gitteret, kan man også finne den private nøkkelen.

To signaturkandidater er delvis basert på *Short integer solution*-problemet (SIS). La q være et heltall. Gitt en begrensning $\beta > 0$ og et ligningssystem

$$A\vec{x} = 0, \quad A \in \mathbb{Z}_q^{n \times m},$$

finn en løsningsvektor $\vec{x}$ slik at $\|\vec{x}\| < \beta$. Det er mulig å vise at dette problemet med passende parametre er minst like vanskelig som en variant av SVP.

Videre er det flere systemer som bygger variasjoner over Regevs *Learning with errors* (LWE) [22]. Intuisjonen for disse er at de består av store lineære ligningssystemer som har fått introdusert en liten mengde støy.

En instans av LWE-problemet består av en uniformt valg $m \times n$-matrise A og en vektor $\vec{b} = A\vec{s} + \vec{e}$, der $\vec{s}$ er en hemmelig vektor valgt fra den uniforme fordelingen over $\mathbb{Z}_q^n$ og $\vec{e}$ er en feilvektor valgt fra en smal diskret normalfordeling rundt 0. I likhet med Diffie-Hellman- problemet kommer LWE

i to varianter: Enten skal man avgjøre om hvorvidt $(A, \vec{b})$ er valgt uniformt eller om det er en ekte instans av LWE, eller finne $\vec{s}$ fra $(A, \vec{b})$. Kandidatene vi ser på her bruker grunnproblemet, samt *Module learning with errors* (MLWE) og *Module learning with rounding* (MLWR), som er spesielle varianter som bruker noe mer underliggende algebraisk struktur, og – for MLWR – velger $\vec{b}$ på en litt annen måte enn beskrevet over. Vi henviser til originalarbeidene for ytterlige detaljer [5, 16, 10].

2.2.2 Dekodingsproblemet

Noen av de mest solide kandidatene er basert på dekodingsproblemet. Det kan på overflaten minne om gitterproblemene over, siden det handler om å finne en omtrentlig løsning på et lineært ligningssystem. Mer presist, la G være en matrise, og $\vec{r}$, $\vec{m}$ og $\vec{e}$ vektorer av passende definisjoner, slik at $\vec{r} = G\vec{m} + \vec{e}$. Gitt $\vec{r}$ og G, finn $\vec{m}$ slik at antallet ikke-null elementer i $\vec{e}$ er minimalt.

Utfordringen for forslagsstillerne er å lage G på en slik måte at det er enkelt å finne m for den som har den hemmelige nøkkelen, men vanskelig for alle andre. Systemer basert på denne ideen har eksistert siden 1978 [18], men har ikke kunnet konkurrert mot for eksempel RSA eller Diffie-Hellman på effektivitet. Også i denne standardiseringsprosessen er de blant de minst effektive, men fordi de er basert på teknikker som har mottatt tiår med oppmerksomhet regner vi det som sannsynlig at Classic McEliece blir standardisert.

2.2.3 Multivariate ligninger

I de to forrige avsnittene prøver man å gjøre lineære ligningssystemer vanskelige å løse ved å innføre støy eller feil. Ved å basere et kryptosystem på multivariate ligninger tar man også konsekvensen av at datamaskiner kan løse lineære ligningssystemer raskt, her ved å i stedet bruke kvadratiske ligningssystemer.

2.2.4 Isogenier

I gruppen av rasjonale punkter på elliptiske kurver kan man enkelt løse diskret logaritme-problemet ved hjelp av Shors algoritme. I kandidaten SIKE forsøker man heller å bli enige om en felles hemmelighet ved å beregne nye kurver i stedet for nye punkter.

Til en elliptisk kurve E kan man tilordne en verdi j som er uavhengig av representasjonen av E, og som derfor kalles j-invarianten. Alle isomorfe kurver har samme j-invariant.

Vi definerer ikke en isogeni formelt her, men nøyer oss med å si at det er en avbildning $f : E_1 \to E_2$, der E_1, E_2 er kurver over samme kropp k, slik at identitetspunktet på E_1 avbildes på identitetspunktet på E_2. Avbildningen er surjektiv, og gruppene av rasjonale punkter på E_1 og E_2 inneholder like mange punkter. Dersom det finnes en isogeni mellom to kurver kalles de isogene. To isogene kurver trenger ikke ha samme j-invariant.

Ideen er videre at hver av partene kan anvende private isogenier ϕ_A, ϕ_B på en startkurve E, og at disse avbildningene kommuterer opp til isomorfi, slik at kurvene $E_{AB} = \phi_B(E_A) = (\phi_B \circ \phi_A)(E) \simeq (\phi_A \circ \phi_B)(E) = E_{BA}$ har samme j-invariant. Den kan igjen brukes til å generere en delt nøkkel.

Beregningsproblemet er å finne isogenien ϕ gitt kurvene E og $\phi(E)$.

2.3 Mikrokontrollere

Det er nyttig å trekke en parallell til Internet of Things (IoT). I løpet av få år har det blitt vanligere at varmeovner, kjøleskap, værstasjoner og blomsterpotter kan kobles sammen i nettverk. I hver slik enhet står det en liten prosessor og kommunikasjonsmodul. Av budsjetthensyn bør slike være så billige som mulige, og akkurat kraftige nok til å gjøre oppgavene sine. Mikrokontrollere kan også utføre kritiske oppgaver i blant annet luftfartøy, romfartøy, båter og kjøretøy, og i medisinsk utstyr. I medisin brukes mikrokontrollere i eksempelvis pacemakere og proteser, og også i større maskiner som røntgenmaskiner. Vi forventer – og ser – en lignende utvikling også i det militære domenet. Det er tidligere vist at sikkerheten kan være svak i IoT [25]. For kritiske anvendelser er det helt avgjørende at sikkerheten forblir sterk til tross for begrensede ressurser.

Målet med vårt eksperiment er å utføre en kvantesikker nøkkelutveksling mellom to mikrokontrollere. Vi har brikken Teensy 3.6 med en 32 bit ARM Cortex M4-prosessor som kjører på 180 MHz. Denne prosessoren er liten og billig med 256 kB RAM, og er vanlig å bruke på små enheter. Nøkkelutvekslingen benytter en signatur- og en KEM-algoritme fra NIST sin standardiseringsprosess. Kjøpt enkeltvis koster disse prosessorene alene godt under 100 kr per stk.

Den enklest tilgjengelige kommunikasjonskanalen til og fra utviklingsbrett med slike mikrokontrollere er protokollen *Universal Asynchronous Receiver/Transmitter* (UART). Den gjør det mulig å utpeke to fysiske kontakter på maskinvaren som inn- og utport, og skrive og lese datastrømmer til disse kontaktene.

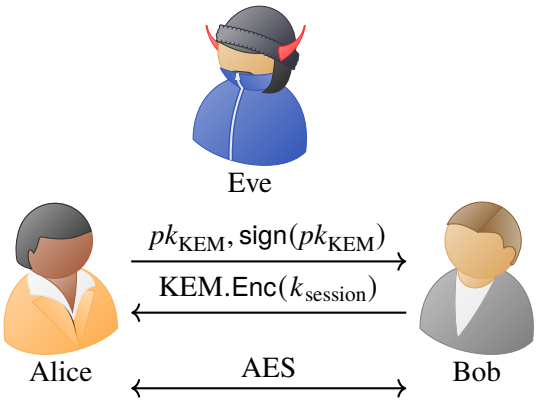
3 Implementasjon

Vi har implementert en demonstrasjon av algoritmene slik at hver funksjonalitet brukes nøyaktig én gang. Figur 3.1 viser protokollen vi har implementert. For enkelhets skyld har vi implementert to klienter; en «initiator» og en «responder». Alice har rollen som initiator, og Bob har rollen som responder. De representerer hver sin mikrokontroller. Alice har ansvar for å starte kommunikasjonen. Hun genererer to nøkkelpar; ett for signatur-protokollen og ett for KEM-protokollen. Så sender hun den offentlige verifikasjonsnøkkelen for signatursystemet og en signert offentlig KEM-nøkkel til Bob. Bob verifiserer signaturen, og bruker den offentlige KEM-nøkkelen til å generere sesjonsnøkkelen og innkapsle denne som en chiffertekst. Chifferteksten sendes til Alice. Hun bruker den private KEM-nøkkelen til å avkapsle chifferteksten slik at hun kan lese sesjonsnøkkelen. Da har begge sesjonsnøkkelen, og de er klare til å bruke denne til AES-kryptert kommunikasjon, noe vi har eksemplifisert gjennom meldingsutveksling.

Vår demonstrasjonsprotokoll er usikker mot en angriper som stiller seg mellom Alice og Bob. En slik motstander vil blant annet kunne utnytte disse punktene:

- Verifiseringsnøkkelen ble sendt sammen med den offentlige KEM-nøkkelen. I en sikker infrastruktur skulle denne vært distribuert på en måte som garanterte at den kom fram uendret, men ikke nødvendigvis hemmelig.

- Bobs svar med en innkapslet sesjonsnøkkel mangler autentisering i vår protokoll. Det kunne vært løst ved at Bob også genererte nøkler for et signatursystem og distribuerte disse på en betryggende måte. Det hadde imidlertid ikke gitt dette eksperimentet noe ytterligere merverdi.



Figur 3.1 Demonstrasjonsprotokoll mellom Alice og Bob. Her er pk_{KEM} den offentlige nøkkelen for KEM-systemet, og $\mathsf{sign}(pk_{KEM})$ er signaturen på nøkkelen. Responsen $KEM.\mathsf{Enc}(k_{session})$ er en symmetrisk nøkkel som er pakket inn ved hjelp av KEM-systemet ved hjelp av pk_{KEM}.

Algoritmene er implementert i biblioteket PQM4 (Post-Quantum M4) [15]. Her ligger det implementasjoner av mange NIST-kandidater for både KEM og signaturer, og implementasjonene er spesielt tilpasset Cortex M4-arkitekturen.

Etter nøkkelutvekslingen demonstrerer Teensyene bruken av sesjonsnøkkelen med AES-kryptert kommunikasjon. Vi bruker AES gjennom biblioteket arduinolibs [27]. Teensy er kompatibel med Arduinos kildekode gjennom en tilleggspakke kalt Teensyduino. Fordelen med å benytte Arduino er at det gir enkle verktøy for kommunikasjon mellom enhetene og mellom enhetene og datamaskiner. Vår innsats i dette prosjektet har vært å flette disse tre bibliotekene sammen og kjøre tester på ytelsen til algoritmene. Pseudokode som viser de overordnede funksjonalitetene er listet i figur 3.2 og 3.3.

Nøklene sendes med Arduinos UART-grensesnitt. Vår kode bruker Serial1, som refererer til pinne 0 og 1 på teensybrettet. Trafikken vår er større enn det UART-grensesnittet normalt er tiltenkt, fra 0,7 kB til 15,4 kB. Dette ble et problem fordi bufferne som skriver meldingene ble fylt opp, og deler av nøklene ble dermed overskrevet før de nådde fram. Det ble spesielt et problem når vi sendte nøklene trådløst, i vårt tilfelle via APC220-antenner. Derfor måtte nøklene deles opp i mindre biter som sendes stykkevis. Til dette laget vi en funksjon `burst_send_string`, som sender et angitt antall bytes før den tar en kort pause, slik at motparten kan prosessere bufferen sin.

Vi har laget en makefil som kompilerer og linker de ulike delene sammen. Den er tilpasset Linux-baserte operativsystemer, men bør uten altfor mye arbeid kunne konverteres til andre operativsystemer også. Byggemiljøet kan enten kobles opp mot en Teensyduino-installasjon, eller mot en frittstående installasjon av verktøykjeden `gcc-arm-eabi-none`. Legg da spesielt merke til filplasseringene som er angitt i `CORE_BASE`, `GCC_BASE` og `UPL_PJRC_B`, som kan tilpasses etter plasseringen av Arduinofiler på den enkelte maskin. For konkret eksperimentering henviser vi til

```
set up serial interfaces (0, 1)
generate KEM keypair (pk_KEM, sk_KEM)
generate signature keypair (pk_SIGN, sk_SIGN)
set pk_KEM||s to SIGN.(Sign)_{sk_SIGN}(pk_KEM)
send (pk_SIGN, pk_KEM||s) to responder

wait for c from responder on serial 1
set k to KEM.Dec(c)
initiate AES-GCM with k

repeat:
  wait for input on serial interfaces
  on input from responder:
    decrypt
    print to serial 0
  on input from serial 0:
    encrypt
    print to serial 1
```

Figur 3.2 Pseudokode for initiator-rollen.

```
set up serial interfaces (0, 1)
wait for (pk_SIGN, pk_KEM||s) from initiator on serial 1
if not SIGN.Verify_{pk_SIGN}(pk_KEM||s):
  halt
set k to 32 random bytes
set c to KEM.Enc_{pk_KEM}(k)
send c to initiator on serial 1
initiate AES-GCM with k

repeat:
  wait for input on serial interfaces
  on input from responder:
    decrypt
    print to serial 0
  on input from serial 0:
    encrypt
    print to serial 1
```

Figur 3.3 Pseudokode for responder-rollen.

Nivå	System	keypair	enc	dec	Fullt navn
1	KYBER	3	4	3	kyber512
	NTRU	998	6	7	ntruhps2048677
	NTRU	1066	3	8	ntruhrss701
	SABER	3	5	5	lightsaber
	FrodoKEM	463	455	451	frodokem640shake
	SIKE	270	440	470	sikep434
2	SIKE	379	619	662	sikep503
3	KYBER	5	6	6	kyber768
	NTRU	1462	7	10	ntruhps4096821
	SABER	6	8	9	saber
	FrodoKEM	1005	1009	1000	frodokem976shake
	NTRU Prime	2186	5	7	sntrup761
	NTRU Prime	1736	9	11	ntrulpr761
	SIKE	673	1234	1241	sikep610
5	KYBER	9	10	9	kyber1024
	SABER	10	13	14	firesaber
	SIKE	1178	1918	2059	sikep741

Tabell 4.1 Tidsmålinger på KEM-variantene. Alle tider i millisekunder. NTRU og NTRU Prime kommer i flere varianter med litt ulike karakteristikker.

koden som er offentlig postet på GitHub[3] og dokumentasjonen som følger med den. Testene er kjørt med kompilatoropsjonen `-O2`, som gir raskest mulig kjøretid uten å øke størrelsen på det kompilerte programmet. Legg merke til at bruk av kompilatoroptimalisering kan gi en sidekanal, for eksempel ved at en algoritme som skulle kjøre i konstant tid i stedet blir raskere, men med kjøretid avhengig av dataene den behandler. Implementasjonene vi bruker er ikke laget med tanke på beskyttelse mot sidekanalangrep, så dette vil være et viktig tema for framtidig arbeid.

For å endre hvilke algoritmer man ønsker å teste må man endre `Makefile` og `src/params.h`. De tilgjengelige algoritmene ligger i `pqm4/crypto_kem` og `pqm4/crypto_sign`.

4 Resultater

Vi har testet KEM-finalistene CRYSTALS-KYBER, NTRU, SABER, FrodoKEM, NTRU Prime og SIKE med forskjellige parametre. Classic McEliece, BIKE og HQC er utelatt fra PQM4-biblioteket fordi den offentlige nøkkelen er for stor for Teensy 3.6. Signatur-finalistene som vi har testet er CRYSTALS-DILITHIUM og FALCON. Rainbow er utelatt av PQM4-biblioteket fordi verifiseringsnøkkelen er for stor. Heller ikke GeMSS, Picnic eller SPHINCS+ finnes i PQM4-biblioteket.

Vi minner om at hver algoritme har ulike parametersett. Parametersettene er tilpasset for å være

[3] `https://github.com/ForsvaretsForskningsinstitutt/Paper-teensy-post-quantum`

Nivå	System	Off. nøkkel	Privat nøkkel	Chiffertekst	Fullt navn
1	KYBER	800	1632	736	kyber512
	NTRU	930	1234	930	ntruhps2048677
	NTRU	1138	1450	1138	ntruhrss701
	SABER	672	1568	736	lightsaber
	FrodoKEM	9616	19888	9720	frodokem640shake
	SIKE	330	374	346	sikep434
2	SIKE	378	434	402	sikep503
3	KYBER	1184	2400	1088	kyber768
	NTRU	1230	1590	1230	ntruhps4096821
	SABER	992	2304	1088	saber
	FrodoKEM	15632	31296	15744	frodokem976shake
	NTRU Prime	1158	1763	1039	sntrup761
	NTRU Prime	1039	1294	1167	ntrulpr761
	SIKE	462	524	486	sikep610
5	KYBER	1568	3168	1568	kyber1024
	SABER	1312	3040	1472	firesaber
	SIKE	564	644	596	sikep751

Tabell 4.2 Minneforbruk i byte for KEM-kandidater.

Nivå	System	keypair	sign	open	Fullt navn
1	DILITHIUM	9	24	9	dilithium2
	FALCON	1049	233	4	falcon-512
	FALCON	1136	110	4	falcon-512-tree
2	DILITHIUM	13	23	14	dilithium3
3	DILITHIUM	18	30	19	dilithium4
4/5	FALCON	1760	505	7	falcon-1024

Tabell 4.3 Tidsmålinger på signaturvariantene. Alle tider i millisekunder.

Nivå	System	Off. nøkkel	Privat nøkkel	Overhead	Fullt navn
1	DILIITHIUM	1184	2800	2044	dilithium2
	FALCON	1281	897	690	falcon-512
	FALCON	57344	897	690	falcon-512-tree
2	DILITHIUM	1472	3504	2701	dilithium3
3	DILITHIUM	1760	3856	3366	dilithium4
4/5	FALCON	1793	2305	1330	falcon-1024

Tabell 4.4 Minneforbruk i byte for signaturkandidater.

kvalifisert til ett av NIST sine sikkerhetsnivåer. Sikkerhetsnivåene er på en skala fra 1 til 5, der 5 er det høyeste sikkerhetsnivået. Et parametersett på nivå 1 vil typisk bruke mindre tid og lagringsplass enn et parametersett på høyere nivå.

Tabell 4.1 viser hvor lang tid delalgoritmene `crypto_kem_keypair`, `crypto_kem_enc` og `crypto_kem_dec` bruker. Tiden er gitt i millisekunder (ms). Funksjonen `crypto_kem_keypair` genererer en en privat nøkkel og en offentlig nøkkel. Funksjonen `crypto_kem_enc` bruker den offentlige nøkkelen til å innkapsle (encapsulate) sesjonsnøkkelen til en chiffertekst, mens funksjonen `crypto_kem_dec` bruker den private nøkkelen til å avkapsle (decapsulate) chifferteksten for å kunne lese sesjonsnøkkelen.

Grundigere benchmarkresultater målt i klokkesykluser finnes distribuert sammen med PQM4. Vi har valgt å bruke konkret tid for å gjøre resultatene mer tilgjengelige for ikke-tekniske lesere.

Tabell 4.2 viser størrelsen på KEM-nøklene og -chiffertekstene. Størrelsene er gitt i byte (B).

Tabell 4.3 viser tidsforbruket til de tre algoritmene `crypto_sign_keypair`, `crypto_sign` og `crypto_sign_open`, oppgitt i millisekunder. Funksjonen `crypto_sign_keypair` genererer en signeringsnøkkel og en verifiseringsnøkkel. Signeringsnøkkelen brukes i funksjonen `crypto_sign` for å signere en melding.

Til sist viser tabell 4.4 størrelsen på signaturnøklene, samt den ekstra størrelsen signaturen gir meldingen.

5 Konklusjon

Vi vet ikke når en tilstrekkelig kraftig kvantedatamaskin vil bli laget, men fordi det *kan* skje i løpet av de neste tiårene må vi ta hensyn til det i dag. Den hyggelige bieffekten er at Forsvaret og det offisielle Norge vil få flere verktøy tilgjengelig for å sikre data og kommunikasjon. Å demonstrere kvantesikker nøkkelutveksling på mikrokontrollene har vært en suksess. Vår demo er ikke en komplett nøkkelutvekslingsprotokoll. Den viser gjennomførbarhet ved at vi demonstrerer hvordan mikrokontrollene kan ta i bruk alle funksjonene fra både signatur- og KEM-systemene, og at de utfører en tilnærmet realistisk nøkkelutveksling. Nøkkelutvekslingen er ikke helt komplett fordi vi kun demonstrerer bruken av signaturer en gang, og ikke overalt der det er nødvendig. De offentlige nøklene måtte i tillegg vært distribuert på forhånd for å komme nær en sikker protokoll.

Vi har testet alle NIST-finalistene fra tredje runde som bruker lite nok minne til å få plass på mikrokontrollene. Vi måtte utelate den kodebaserte KEM-protokollen Classic McEliece, og den multivariatbaserte signaturprotokollen Rainbow. Signatur- og KEM-protokollene som gjensto var alle gitterbaserte.

Ved siden av å demonstrere gjennomførbarhet, har vi også målt kjøretidene. Forskjellene er til dels dramatiske, og ikke overraskende er de systemene med de mest konservative valgene rundt sikkerhet også de tregeste i sin klasse, som for eksempel FrodoKEM – som vi ikke kunne kjøre

kategori 5-varianten av – og Classic McEliece, som ikke var tilgjengelig for oss uansett. Målingene tilsier at enkelte av kandidatene vil kunne være praktiske også på svært begrenset maskinvare, men framtidig anvendelse av dem vil først avhenge av at de faktisk blir standardisert, og deretter at de relevante myndighetene godkjenner algoritmene til bruk for gradert informasjon. Om bare de mest konservative algoritmene blir godkjent for graderte systemer vil det for eksempel ha innvirkning på hvilken prosesseringskraft vi må utnytte framtidige autonome enheter med. På sivil side vil sikring av IoT måtte ta noen av de samme avveiingene.

Det er to naturlige steg videre for disse undersøkelsene. Det første er å implementere algoritmene på maskinvare eller programmerbar maskinvare (FPGA) for å måle hastigheten og effektiviteten på slike. En annen retning er å beskytte både programvare- og maskinvareimplementasjoner mot passive og inngripende sidekanalangrep. Parallelt foregår det aktiviteter over hele verden for å undersøke kandidatene med kritisk blikk, og forhåpentligvis nøye nok til at vi kan stole på sikkerheten til de systemene som ender opp med å bli standardisert.

Ordliste

AES *Advanced Encryption Standard*, den mest brukte algoritmen for symmetrisk kryptografi i verden; ble valgt ut som vinner i en åpen konkurranse arrangert/fasilitert av NIST; kryptologer og myndigheter har jevnt over høy tillit til AES når den brukes på riktig måte. 9

angrep et generisk uttrykk for å få et system til å opptre på en hvilken som helst annen måte enn det protokollen eller definisjonen tilsier, eller at man kan knekke egenskaper på kortere tid enn det ville tatt å sjekke alle mulige nøkler. 9

asymmetrisk kryptografi et konsept der det er mulig å offentliggjøre en nøkkel, for eksempel for å kryptere eller verifisere en signatur; samtidig kan bare den private nøkkelen brukes til å dekryptere eller signere (jf. symmetrisk kryptografi). 7, 9

chiffertekst en kryptert melding (i kontrast til en klartekst). 14

diskret logaritme et underliggende problem for mange kryptografiske teknikker; gitt en gruppe G og elementer g, $h = g^a$, så skal man beregne a fra g og h, noe som i mange tilfeller er vanskelig for en datamaskin, men alltid enkelt for en kvantedatamaskin. 9

faktorisering et underliggende problem for mange kryptosystemer, fordi det regnes som vanskelig for en datamaskin å faktorisere store tall, samtidig som det er enkelt å multiplisere store tall; en kvantedatamaskin vil kunne faktorisere raskt, og dermed faller alle kryptosystemer som er basert på faktorisering sammen, f.eks. RSA. 9

Grovers algoritme en algoritme for kvantedatamaskiner som gjør det mulig å søke gjennom en uordnet mengde med n elementer ved å bare bruke omtrent $\sqrt{n}$ tid; den teoretiske konsekvensen er at alle kryptosystemer må kvadrere nøkkelrommet sitt (som betyr å doble nøkkellengden) for å oppnå samme sikkerhet mot kvantedatamaskiner som de tidligere hadde mot klassiske datamaskiner; i praksis trengs det nok mindre økning, men det er nå vanlig å bruke 256 bit-nøkler i AES, mot 128 bit tidligere. 9

hashfunksjon en funksjon som komprimerer vilkårlig lang input til en kort streng av forhåndsdefinert lengde; vi forventer at små endringer i input gir store endringer i output, gitt en gyldig output skal det være vanskelig å finne tilsvarende input, gitt en melding m som gir hash h skal det i praksis være umulig å finne en annen melding som gir samme hash, og det skal være like umulig å finne to meldinger som gir samme hash, uansett hvilken. 10

IoT *Internet of Things*, eller tingenes internett; en betegnelse på det at svært mange små enheter som sensorer, droner, varmeovner og brødristere er tilkoblet internett, og der enhetene snakker sammen uten at det nødvendigvis er mennesker med i kommunikasjonen. 12

kvantedatamaskin en datamaskin som opererer på *kvantebit*; elementer som ikke er 0 eller 1, men som eksisterer i en superposisjon mellom begge; posisjonen avhenger av en sannsynlighetsfordeling som kan manipuleres, og når flere slike kvantebit er sammenkoblet kan man manipulere sannsynlighetsfordelingen til hele tilstanden. 7

kvantesikker et kryptosystem er kvantesikkert dersom det ikke eksisterer effektive algoritmer for kvantedatamaskiner som kan knekke systemet; et system kan være antatt sikkert i den klassiske modellen – mot vanlige datamaskiner – men ikke mot kvantedatamaskiner, som f.eks. RSA. 13

NIST *National Institute of Standards and Technology*, en amerikansk sivil institusjon som har gjennomført flere standardiseringsprosesser for kryptografi der både algoritmer og kriterier har vært transparente. 8, 9

RSA fra opphavsmennene Rivest, Shamir og Adleman, som i 1977 lanserte det første kryptosystemet med en offentlig nøkkel; RSA slik den ble presentert regnes i dag som usikker, men varianter som pakker inn meldingen i tilfeldighet er fortsatt i bruk, spesielt som signatursystem. 9

Shors algoritme en algoritme for kvantedatamaskiner som blant annet gjør det mulig å faktorisere store tall og løse diskrete logaritmer på kort tid; hovedideen er å finne sykluser i den underliggende matematiske strukturen ved hjelp av verktøy som ikke er tilgjengelige for vanlige datamaskiner. 9

sidekanal selv om en algoritme er vanskelig å knekke direkte kan man bruke karakteristika ved algoritmen til å forstå hvilke data den behandler, for eksempel kan en implementasjon bruke litt mer tid eller strøm hver gang den behandler en 1 enn når den behandler en 0, og ved å måle veldig presist kan man da bruke en slik sidekanal til å finne informasjon om de hemmelige dataene. 14

signatur data som kobler innholdet i en pakke eller dokument til en offentlig nøkkel og potensielt en identitet; kan brukes til å gi meldingen autentisitet. 10

symmetrisk kryptografi et konsept der to eller flere parter sitter med identisk, hemmelig nøkkelmateriale eller utfører de samme algoritmene (jf. asymmetrisk kryptografi). 7, 9

Referanser

[1] Carlos Aguilar Melchor, Nicolas Aragon, Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, Edoardo Persichetti og Gilles Zémor. *HQC*. Tekn. rapp. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions. National Institute of Standards og Technology, 2019.

[2] Miklós Ajtai. «The Shortest Vector Problem in L2 is NP-hard for Randomized Reductions (Extended Abstract)». I: *30th ACM STOC*. Dallas, TX, USA: ACM Press, mai 1998, s. 10–19. DOI: 10.1145/276698.276705.

[3] Gorjan Alagic, Jacob Alperin-Sheriff, Daniel Apon, David Cooper, Quynh Dang, John Kelsey, Yi-Kai Liu, Carl Miller, Dustin Moody, Rene Peralta, Ray Perlner, Angela Robinson og Daniel Smith-Tone. *Status Report on the Second Round of the NIST Post-Quantum Cryptography Standardization Process*. NISTIR 8309. https://doi.org/10.6028/NIST.IR.8309. NIST, jul. 2020.

[4] Nicolas Aragon, Paulo Barreto, Slim Bettaieb, Loic Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Phillipe Gaborit, Shay Gueron, Tim Guneysu, Carlos Aguilar Melchor, Rafael Misoczki, Edoardo Persichetti, Nicolas Sendrier, Jean-Pierre Tillich, Gilles Zémor og Valentin Vasseur. *BIKE*. Tekn. rapp. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions. National Institute of Standards og Technology, 2019.

[5] Abhishek Banerjee, Chris Peikert og Alon Rosen. «Pseudorandom Functions and Lattices». I: *EUROCRYPT 2012*. Red. av David Pointcheval og Thomas Johansson. Bd. 7237. LNCS. Cambridge, UK: Springer, Heidelberg, Germany, apr. 2012, s. 719–737. DOI: 10.1007/978-3-642-29011-4_42.

[6] Daniel J. Bernstein, Tung Chou, Tanja Lange, Ingo von Maurich, Rafael Misoczki, Ruben Niederhagen, Edoardo Persichetti, Christiane Peters, Peter Schwabe, Nicolas Sendrier, Jakub Szefer og Wen Wang. *Classic McEliece*. Tekn. rapp. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions. National Institute of Standards og Technology, 2019.

[7] Daniel J. Bernstein, Chitchanok Chuengsatiansup, Tanja Lange og Christine van Vredendaal. *NTRU Prime*. Tekn. rapp. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions. National Institute of Standards og Technology, 2019.

[8] A. Casanova, J.-C. Faugère, G. Macario-Rat, J. Patarin, L. Perret og J. Ryckeghem. *GeMSS*. Tekn. rapp. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions. National Institute of Standards og Technology, 2019.

[9] Jan-Pieter D'Anvers, Angshuman Karmakar, Sujoy Sinha Roy og Frederik Vercauteren. *SABER*. Tekn. rapp. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions. National Institute of Standards og Technology, 2019.

[10] Jan-Pieter D'Anvers, Angshuman Karmakar, Sujoy Sinha Roy og Frederik Vercauteren. «Saber: Module-LWR Based Key Exchange, CPA-Secure Encryption and CCA-Secure KEM». I: *AFRICACRYPT 18*. Red. av Antoine Joux, Abderrahmane Nitaj og Tajjeeddine Rachidi. Bd. 10831. LNCS. Marrakesh, Morocco: Springer, Heidelberg, Germany, mai 2018, s. 282–305. DOI: 10.1007/978-3-319-89339-6_16.

[11] Jintai Ding, Ming-Shing Chen, Albrecht Petzoldt, Dieter Schmidt og Bo-Yin Yang. *Rainbow*. Tekn. rapp. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions. National Institute of Standards og Technology, 2019.

[12] Lov K. Grover. «A Fast Quantum Mechanical Algorithm for Database Search». I: *28th ACM STOC*. Philadephia, PA, USA: ACM Press, mai 1996, s. 212–219. DOI: 10.1145/237814.237866.

[13] Andreas Hulsing, Daniel J. Bernstein, Christoph Dobraunig, Maria Eichlseder, Scott Fluhrer, Stefan-Lukas Gazdag, Panos Kampanakis, Stefan Kolbl, Tanja Lange, Martin M Lauridsen, Florian Mendel, Ruben Niederhagen, Christian Rechberger, Joost Rijneveld, Peter Schwabe og Jean-Philippe Aumasson. *SPHINCS+*. Tekn. rapp. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions. National Institute of Standards og Technology, 2019.

[14] David Jao, Reza Azarderakhsh, Matthew Campagna, Craig Costello, Luca De Feo, Basil Hess, Amir Jalali, Brian Koziel, Brian LaMacchia, Patrick Longa, Michael Naehrig, Joost Renes, Vladimir Soukharev, David Urbanik og Geovandro Pereira. *SIKE*. Tekn. rapp. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions. National Institute of Standards og Technology, 2019.

[15] Matthias J. Kannwischer, Joost Rijneveld, Peter Schwabe og Ko Stoffelen. *PQM4: Post-quantum crypto library for the ARM Cortex-M4*. https://github.com/mupq/pqm4.

[16] Adeline Langlois og Damien Stehlé. *Worst-Case to Average-Case Reductions for Module Lattices*. Cryptology ePrint Archive, Report 2012/090. http://eprint.iacr.org/2012/090. 2012.

[17] Vadim Lyubashevsky, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Peter Schwabe, Gregor Seiler og Damien Stehlé. *CRYSTALS-DILITHIUM*. Tekn. rapp. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions. National Institute of Standards og Technology, 2019.

[18] Robert McEliece. «A Public-Key Cryptosystem Based On Algebraic Coding Theory». I: *DSN Progress Report 42-44* (1978), s. 114–116.

[19] Michael Naehrig, Erdem Alkim, Joppe Bos, Léo Ducas, Karen Easterbrook, Brian LaMacchia, Patrick Longa, Ilya Mironov, Valeria Nikolaenko, Christopher Peikert, Ananth Raghunathan og Douglas Stebila. *FrodoKEM*. Tekn. rapp. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions. National Institute of Standards og Technology, 2019.

[20] NIST. *Submission Requirements and Evaluation Criteria for the Post-Quantum Cryptography Standardization Process*. https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/call-for-proposals-final-dec-2016.pdf (accessed 2020-08-16). 2016.

[21] Thomas Prest, Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Ricosset, Gregor Seiler, William Whyte og Zhenfei Zhang. *FALCON*. Tekn. rapp. available at `https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions`. National Institute of Standards og Technology, 2019.

[22] Oded Regev. «On lattices, learning with errors, random linear codes, and cryptography». I: *37th ACM STOC*. Red. av Harold N. Gabow og Ronald Fagin. Baltimore, MA, USA: ACM Press, mai 2005, s. 84–93. DOI: `10.1145/1060590.1060603`.

[23] Peter Schwabe, Roberto Avanzi, Joppe Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Gregor Seiler og Damien Stehlé. *CRYSTALS-KYBER*. Tekn. rapp. available at `https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions`. National Institute of Standards og Technology, 2019.

[24] Peter W. Shor. «Algorithms for Quantum Computation: Discrete Logarithms and Factoring». I: *35th FOCS*. Santa Fe, NM, USA: IEEE Computer Society Press, nov. 1994, s. 124–134. DOI: `10.1109/SFCS.1994.365700`.

[25] Martin Strand og Jan Henrik Wiik. *Kryptografisk sikring av autonome og ubemannede enheter – eksisterende forskning*. FFI-rapport 19/02042. FFI, 2019.

[26] P. van Emde-Boas. *Another NP-complete partition problem and the complexity of computing short vectors in a lattice*. Report. Department of Mathematics. University of Amsterdam. Department of Matematics, Univeristy of Amsterdam, 1981.

[27] Rhys Weatherley. *Arduino Cryptography Library*. https://github.com/rweather/arduinolibs.

[28] Greg Zaverucha, Melissa Chase, David Derler, Steven Goldfeder, Claudio Orlandi, Sebastian Ramacher, Christian Rechberger, Daniel Slamanig, Jonathan Katz, Xiao Wang og Vladmir Kolesnikov. *Picnic*. Tekn. rapp. available at `https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions`. National Institute of Standards og Technology, 2019.

[29] Zhenfei Zhang, Cong Chen, Jeffrey Hoffstein, William Whyte, John M. Schanck, Andreas Hulsing, Joost Rijneveld, Peter Schwabe og Oussama Danba. *NTRUEncrypt*. Tekn. rapp. available at `https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions`. National Institute of Standards og Technology, 2019.

About FFI

The Norwegian Defence Research Establishment (FFI) was founded 11th of April 1946. It is organised as an administrative agency subordinate to the Ministry of Defence.

FFI's MISSION

FFI is the prime institution responsible for defence related research in Norway. Its principal mission is to carry out research and development to meet the requirements of the Armed Forces. FFI has the role of chief adviser to the political and military leadership. In particular, the institute shall focus on aspects of the development in science and technology that can influence our security policy or defence planning.

FFI's VISION

FFI turns knowledge and ideas into an efficient defence.

FFI's CHARACTERISTICS

Creative, daring, broad-minded and responsible.

Om FFI

Forsvarets forskningsinstitutt ble etablert 11. april 1946. Instituttet er organisert som et forvaltningsorgan med særskilte fullmakter underlagt Forsvarsdepartementet.

FFIs FORMÅL

Forsvarets forskningsinstitutt er Forsvarets sentrale forskningsinstitusjon og har som formål å drive forskning og utvikling for Forsvarets behov. Videre er FFI rådgiver overfor Forsvarets strategiske ledelse. Spesielt skal instituttet følge opp trekk ved vitenskapelig og militærteknisk utvikling som kan påvirke forutsetningene for sikkerhetspolitikken eller forsvarsplanleggingen.

FFIs VISJON

FFI gjør kunnskap og ideer til et effektivt forsvar.

FFIs VERDIER

Skapende, drivende, vidsynt og ansvarlig.

FFIs organisasjon