Lattice cryptography: isomorphisms & dual attacks

Pulles, L.N.

Samenvatting

Dit proefschrift getiteld “Rooster Cryptografie: Isomorfismes & Duale Aanvallen” gaat over roosters. Veel van de cryptografie die nu gebruikt wordt, kan in de toekomst met (grote) quantumcomputers worden gekraakt! Voor roosters bestaan er diverse berekeningsproblemen, die makkelijk zijn in 2 of 3 dimensies, maar moeilijk zijn in hoge dimensies. Deze problemen zijn, voor zover bekend, moeilijk voor computers *én* quantumcomputers. Daarom kunnen we met roosters *post-quantumcryptografie* bouwen, wat veilig blijft zelfs als er grote quantumcomputers zijn.

Cryptografie heeft als doel om communicatie tussen twee of meer partijen veilig te laten verlopen. Twee belangrijke cryptografische protocollen zijn *versleuteling* en *handtekeningen*. Versleuteling maakt de inhoud van berichten onbegrijpbaar voor afluisteraars. Door een handtekening valt het te controleren of iemand echt een bepaald bericht heeft gestuurd. In een handtekening protocol bezitten beide partijen verschillende sleutels, vergelijkbaar met de echte wereld: één partij schrijft handtekeningen met een bijzondere pen, en de ander weet de unieke kleursamenstelling van de inkt uit de pen. Versleuteling gaat vergelijkbaar: iedereen kan een bericht versleutelen met een *openbare sleutel*, maar alleen degene met de *geheime sleutel* kan het ontcijferen.

Roosters kunnen worden gebruikt om zo’n openbaar/geheim sleutelpaar te maken. De geheime sleutel is een beschrijving van het rooster, die bestaat uit de kortste roosterpunten. Deze sleutel is geheim omdat het in hoge dimensie moeilijk is om überhaupt één redelijk kort roosterpunt te berekenen. De openbare sleutel is een willekeurige beschrijving van het rooster met lange roosterpunten. Het snelste algoritme om de geheime sleutel te geven die hoort bij een openbare sleutel, heeft tijd en geheugen nodig dat exponentieel groeit in de dimensie.

Deel II gaat over de *duale aanval*. Dit is een algoritme wat als invoer een roosterbeschrijving en een punt krijgt, en dan met behulp van het *duaal rooster* het roosterpunt vindt wat er het dichtste bij ligt. Elk duaal roosterpunt correspondeert met een golffunctie (karakter) die in elk roosterpunt maximaal is. Met deze golffunctie kun je bepalen of een punt dicht bij het rooster ligt. Door het rooster op te delen als een schaakbord in een zwart deelrooster en een witte nevenklasse, kun je bepalen of het zwarte deelrooster dichter bij het punt ligt dan het witte. Als je deze methode vaak genoeg herhaalt, weet je uiteindelijk welk roosterpunt het dichtste bij het punt ligt.

Hoofdstuk 3 toont aan dat een in de literatuur gebruikte aanname voor het analyseren van de zogenaamde *duale-zeefaanval* fout is, want de aanname is tegenstrijdig in verschillende theoretische situaties, en uit experimenten blijkt deze verkeerde voorspelling te geven. Hoofdstuk 4 bevat een alternatieve aanname voor de duale zeefaanval. Deze aanname leidt tot nauwkeurige voorspellingen voor verschillende belangrijke kansverdelingen van punten.

Deel III gaat over het *rooster isomorfisme probleem* (LIP). Speciale roosters met goede eigenschappen zijn handig om in *handtekeningenschema's* te gebruiken, want deze maken het schema zeer efficiënt. Omdat elke cryptoloog goede beschrijvingen van deze roosters kent, moet het rooster op een andere manier verborgen worden. Met behulp van LIP kunnen we het rooster verbergen, door het in een willekeurige richting te draaien. De draaiing is nu de geheime sleutel.

In Hoofdstuk 5 bouwen we het handtekeningenschema Hawk op basis van LIP. Om de geheime sleutel van Hawk te krijgen, moet je smLIP, een moduulvariant van LIP oplossen voor het simpele rooster $\mathbb{Z}^n$, het rooster van geheeltallige punten. Veiligheid van Hawk is gebaseerd op het nieuwe *nog een korte vector probleem* (omSVP). In Hoofdstuk 6 bestuderen we *automorfismes* van het moduulrooster: een draaiing van de ruimte die het rooster op zichzelf afbeeldt. Stelling 6.2 zegt dat met behulp van een niet-triviaal automorfisme, het veel makkelijker is om korte vectoren te vinden, en om smLIP op te lossen. Een automorfisme stuurt een kort roosterpunt naar een (mogelijk ander) kort roosterpunt, en kan hierdoor omSVP oplossen. Als smLIP een lastig probleem is, dan zal het moeilijk zijn om een niet-triviaal automorfisme te vinden, en zal het moeilijk zijn om omSVP op te lossen door het vinden van een niet-triviaal automorfisme.