

**De invloed van**

**de kwantumcomputer op het RSA-systeem**

De invloed van

de kwantumcomputer op het RSA-systeem

Inhoudsopgave

[Voorwoord 4](#_Toc62506851)

[Samenvatting 5](#_Toc62506852)

[Inleiding 7](#_Toc62506853)

[Onderzoeksplan 8](#_Toc62506854)

[Planning 9](#_Toc62506855)

[Hoe werken kwantumcomputers? 11](#_Toc62506856)

[Kwantummechanica 11](#_Toc62506857)

[Kwantumcomputer 12](#_Toc62506858)

[Algoritmes 13](#_Toc62506859)

[Shor 13](#_Toc62506860)

[Wat is de ontwikkeling van (kwantum) computers? 16](#_Toc62506861)

[Hoe is het RSA-systeem tot stand gekomen? 18](#_Toc62506862)

[Caesar-Cipher 18](#_Toc62506863)

[Atbash-geheimschrift 19](#_Toc62506864)

[Blaise de Vigenère 19](#_Toc62506865)

[Het vernam systeem 20](#_Toc62506866)

[Enigma-code 20](#_Toc62506867)

[Data encryption standard 21](#_Toc62506868)

[Diffie-Hellman 21](#_Toc62506869)

[Het RSA-systeem 21](#_Toc62506870)

[Hoe werkt het RSA-systeem? 23](#_Toc62506871)

[Geschiedenis van het RSA-systeem 23](#_Toc62506872)

[Werking van het RSA-systeem 24](#_Toc62506873)

[Priemgetallen 24](#_Toc62506874)

[Algoritme van Euclides 25](#_Toc62506875)

[Modulo rekenen 27](#_Toc62506876)

[Priemfactoratie met Fermat 29](#_Toc62506877)

[Diffie en Hellman 30](#_Toc62506878)

[Fermat en Euler 31](#_Toc62506879)

[De Euler φ-functie 32](#_Toc62506880)

[Het RSA-algoritme 34](#_Toc62506881)

[In hoeverre is het RSA-systeem veilig? 37](#_Toc62506882)

[Trial division 37](#_Toc62506883)

[Brute force hacking 37](#_Toc62506884)

[Number field sieve 37](#_Toc62506885)

[Wat is de invloed van kwantumcomputers op de cryptografie? 39](#_Toc62506886)

[Huidige encryptiemethoden 39](#_Toc62506887)

[Kwantumcryptografie 40](#_Toc62506888)

[Toekomst 42](#_Toc62506889)

[Conclusie 43](#_Toc62506890)

[Discussie en reflectie 44](#_Toc62506891)

[Aanbeveling voor vervolgonderzoek 45](#_Toc62506892)

[Bronnenlijst 46](#_Toc62506893)

[Bijlagen 50](#_Toc62506894)

[Logboek 51](#_Toc62506895)

[Bijlagen 55](#_Toc62506896)

# Voorwoord

Cryptografie, geheimtaal, hackers, veiligheid en wiskunde, een aantal aanleidingen om een onderzoek te starten waarbij wij informatie verzamelen over hoe geheimtaal en wiskunde in elkaar overlopen. Tess wil graag kunstmatige intelligentie studeren en het leek haar leuk om iets te onderzoeken in deze richting. Tijdens het overleg met May, waren we tot overeenstemming gekomen om ons te verdiepen in cryptografie. Tijdens de PWS-dagen heeft mevrouw … ons geholpen met het opstellen van de hoofdvraag. Daarna hebben we onze deelvragen opgesteld met een bron die wij kregen aangeraden door mevrouw ... De bron is een website die is geschreven door meneer …. Wij hebben hier ruim 80 uur aan besteed. Wij willen vooral meneer Rauwerda bedanken voor de begeleiding die u ons heeft gegeven in deze bijzondere tijd.

# Samenvatting

Dit onderzoek gaat over kwantumcomputers en encryptiemethoden. Wij wilden graag weten wat de invloed was van kwantumcomputers op de encryptie, de hoofdvraag luidt dan ook ‘Wat is de invloed van kwantumcomputers op de veiligheid van het RSA-systeem?’. Onze hypothese hierbij is dat kwantumcomputers de veiligheid op het RSA-systeem verslechteren. De methode die is gebruikt, is het analyseren van bijpassende bronnen. De bronnen die wij hebben gebruikt zijn het internet en boeken over de cryptografie en (kwantum)computers.

Kwantumcomputers maken gebruik van de wetten van de kwantummechanica. De berekeningen worden niet lineair gemaakt maar parallel. Dit wordt bereikt door middel van de eigenschap superpositie, die elementaire deeltjes hebben. De elementaire deeltjes worden, in de kwantumcomputers, kwantum bits genoemd. Door ‘quantum state transfer’ wordt informatie doorgegeven van kwantum bit naar kwantum bit. Een belangrijk kwantumalgoritme voor de krakers van de encryptiemethoden, is het algoritme van Shor. Deze kan getallen ontbinden in priemfactoren.

De kwantumcomputer is nog volop in ontwikkeling. Sinds 1980 is het idee er al, van een kwantumcomputer. In de 30 jaar hierna werden verschillende kwantumprotocollen opgesteld. Alleen in de afgelopen 10 jaar is het enorm snel gegaan. Zo bestaat er sinds 2015 al de eerst werkende kwantumcomputer. Dit zal zich in de aankomende jaren nog sneller ontwikkelen, net zoals de ‘normale’ computer, 35 jaar na de eerst elektronische computer stond er ook een computer thuis op het bureau.

Het RSA-systeem is niet uit het niets tot stand gekomen. De eerst bedachte encryptiemethode was de Caesar Cipher. Het geheim achter deze methode wist men na een korte tijd te achterhalen. Een andere methode dat op de Caesar Cipher lijkt is het Atbash geheimschrift. Nadat de Caesar Cipher werd gekraakt, had Blaise de Vingère een andere methode bedacht die minder eenvoudig te kraken was. In de loop van de 19e eeuw werd het Vernam systeem bedacht. Deze methode werd tegen de Duitsers gebruikt en tijdens de Tweede Wereld Oorlog werd de Enigma Code door de Duitsers gebruikt. In de tweede helft van de 20e eeuw werden Data Encryption Standard en de Diffie-Hellman methode bedacht en deze waren het begin van de ontwikkeling van het RSA-systeem.

Het RSA-systeem is een encryptiemethode die hedendaags vaak wordt gebruikt om onze persoonlijke gegevens veilig te bewaren. Deze methode werkt op een wiskundige wijze. Om te begrijpen hoe dit systeem werkt, is het van belang om te weten wat de volgende wiskundige termen zijn en hoe ze werken: priemgetallen, algoritme van Euclides, modulo rekenen, priemfactoratie met Fermat, Diffie en Hellman en de Euler φ-functie.

Er zijn een aantal decodeer methodes bedacht die de veiligheid van het RSA-systeem in gevaar kunnen brengen. Zo zijn Trial Division en Brute Force Hacking kraakmethodes die het RSA-systeem kunnen kraken, maar het gebruiken van deze methodes duurt erg lang en kan niet werken om een groot getal dat uit 2 grote, onbekende priemfactoren bestaat te ontbinden. Number Field Sieve is het wel gelukt om het RSA-systeem ooit te kraken, maar nadat het aantal bits van het RSA verdubbeld werd, werd dat ook lastig. Het RSA-systeem is nog wel veilig voor de decodeermethodes die uitgevoerd kunnen worden met een normale computer.

Door de opkomst van kwantumcomputers is de encryptie in groot gevaar. De symmetrische encryptiemethoden zijn minder in gevaar dan de asymmetrische encryptiemethoden. Kwantumcomputers kunnen natuurlijk ook gebruikt worden om juist de gegevens te beveiligen, dit wordt dan kwantumencryptie genoemd. Op dit moment zijn kwantumcomputers nog niet sterk genoeg om de huidige computers te vervangen, het is ook nog niet duidelijk wanneer dit gaat gebeuren. Wel is het waarschijnlijk dat er een wedloop gaat ontstaan tussen de ontwikkelaars en de krakers van de encryptiemethoden.

# Inleiding

Cryptografie is een vertakking in de wiskunde waarbij gebruik wordt gemaakt van slim versleutelen. Het doel van cryptografie is het sturen van verborgen boodschappen die alleen te decoderen zijn door de ontvanger zelf. Daarom speelde cryptografie tijdens de Tweede Wereldoorlog een belangrijke rol. Bondgenoten stuurden elkaar geheime brieven met verborgen boodschappen, voor het geval dat de brieven in de verkeerde handen vielen. Cryptografie is in de loop van de tijd verder ontwikkeld en speelt tegenwoordig ook een cruciale rol bij het veilig houden van onze informatie.

Door de opkomst van ver ontwikkelde snelle computers, genaamd kwantumcomputers, is er een mogelijkheid dat onze informatie niet meer zo veilig is als dat we denken. Een kwantumcomputer maakt gebruik van kwantummechanica en kan hierdoor enorm veel data analyseren en veel codes tegelijk invoeren. Dit heeft ervoor gezorgd dat degenen die dit soort computers in het bezit hebben een bedreiging kunnen vormen voor persoonlijke informatie die op internet staat.

Een van de manieren waarop deze informatie kan worden beveiligd is door middel van een encryptiemethode, genaamd het RSA-systeem. Het RSA-systeem staat voor de uitvinders van het RSA namelijk, Ron **R**ivest, Adi **S**hamir en Leonard **A**dleman. Het RSA is een geheimschrift waarbij er twee sleutels zijn die voor meerdere mensen toegankelijk zijn. Er is een privé-sleutel voor de maker en een publieke sleutel die voor iedereen bekend kan worden gemaakt.

Dit klinkt allemaal erg onveilig en makkelijk te kraken. Wij vragen ons daarom nu af of kwantumcomputers hier invloed op uit oefenen. De hoofdvraag die wij hierbij hebben opgesteld is: ‘Wat is de invloed van kwantumcomputers op de veiligheid van het RSA-systeem?’. Om tot een antwoord te komen op onze hoofdvraag, hebben wij een aantal deelvragen opgesteld.

Onze deelvragen zijn:

* Hoe werken kwantumcomputers?
* Wat is de ontwikkeling van (kwantum)computers?
* Hoe is het RSA-systeem tot stand gekomen?
* Hoe werkt het RSA-systeem?
* In hoeverre is het RSA-systeem veilig?
* Wat is de invloed van kwantumcomputers op de cryptografie?

Uit ons vooronderzoek, om onze hoofdvraag en deelvragen te kunnen vormen, hebben wij een hypothese opgesteld. Wij stellen dat kwantumcomputers ervoor zorgen dat de veiligheid van het RSA-systeem daalt.

# Onderzoeksplan

Het onderzoeksplan dat wij hebben opgesteld staat hieronder vermeld. Daarbij vonden wij het zelf fijn om gezamenlijk de hoofdstukken uit te werken en daarbij geen strakke taakverdeling op te stellen. Door de Coronapandemie hebben wij dit plan enkele malen herzien en aangepast waar nodig.

|  |  |
| --- | --- |
| Onderzoeksplan van: May Korkis en Tess de Vries | |
| Onderwerp | Cryptografie |
| Onderzoeksvraag/ ontwerpopdracht  (Hoofdvraag) | Wat is de invloed van kwantumcomputers op de veiligheid van het RSA-systeem? |
| Hypothese  (Indien nodig) | Kwantumcomputers zorgen ervoor dat het RSA-systeem minder veilig wordt. |
| Deelvraag | Hoe werken kwantumcomputers? |
| Deelvraag | Wat is de ontwikkeling van (kwantum)computers? |
| Deelvraag | Hoe is het RSA-systeem tot stand gekomen? |
| Deelvraag | Hoe werkt het RSA-systeem? |
| Deelvraag | In hoeverre is het RSA-systeem veilig? |
| Deelvraag | Wat is de invloed van kwantumcomputers op de cryptografie? |
| Methode | Analyseren van bijpassende bronnen |
| Materiaal  Wat heb je nodig? | - Internet  - Boeken  - Docenten wiskunde en natuurkunde |
| Welke informatie ga je gebruiken? | Informatie over kwantumcomputers, computers en cryptografie. |
| Tijdplanning | Zie hieronder |
| Taakverdeling | Vormgeving werkstuk: Tess  Voorwoord: May en Tess  Samenvatting: May en Tess  Inleiding: May en Tess  Onderzoeksplan: Tess  Deelvraag 1: Tess  Deelvraag 2: Tess  Deelvraag 3: May  Deelvraag 4: May  Deelvraag 5: May  Deelvraag 6: Tess  Conclusie en discussie: May en Tess  Conclusie algemeen: May en Tess  Aanbeveling voor vervolgonderzoek: May en Tess  Bronnenlijst: May en Tess  Logboek: May en Tess |

## Planning

|  |  |  |  |
| --- | --- | --- | --- |
| Planning profielwerkstuk | Activiteit | Verwachte aantal uren per persoon | Afgerond voor |
| Stap 1: oriënteren op het onderwerp | * Inspiratie zoeken op internet * Onderwerp kiezen * Gebruik maken boeken (vb. verborgen boodschappen) | 2 uur | 29/06/2020 |
| Stap 2: richten van onderzoeksvraag | * Verdiepen in het onderwerp * Naar specifieke onderwerpen kijken * Opstellen van hoofdvraag en geschikte deelvragen | 5 uur | 01/07/2020 |
| Stap 3; planning maken | * Taakverdeling opstellen * Nadenken over wat realistisch is * Antwoord op de vraag; Wat moet er precies gedaan worden en wanneer moet dit af zijn? | 2 uur | 03/07/2020 |
| Stap 4: verzamelen van de informatie | * Geschikte bronnen zoeken op internet en in de bibliotheek eventueel ook nog wat musea bezoeken zoals Nemo | 25 uur | 12/11/2020 |
| Stap 5: analyseren en concluderen | * Conceptversie deelvraag 1 * Feedback deelvraag 1 * Deelvraag 1 afronden met bronvermelding en afbeeldingen * Conceptversie deelvraag 2 * Feedback deelvraag 2 * Deelvraag 2 afronden met bronvermelding en afbeeldingen * Conceptversie deelvraag 3 * Feedback deelvraag 3 * Deelvraag 3 afronden met bronvermelding en afbeeldingen * Conceptversie deelvraag 4 * Feedback deelvraag 4 * Deelvraag 4 afronden met bronvermelding en afbeeldingen * Conceptversie deelvraag 5 * Feedback deelvraag 5 * Deelvraag 5 afronden met bronvermelding en afbeeldingen * Conceptversie deelvraag 6 * Feedback deelvraag 6 * Deelvraag 6 afronden met bronvermelding en afbeeldingen * Conceptversie conclusie * Feedback conclusie * Conclusie afronden met afbeeldingen * Inleiding schrijven * Discussie schrijven * Bronnenlijst maken * Omslag, titelpagina en inhoudsopgave maken | 30 uur | 04/01/2021 |
| Stap 6: rapporteren en presenteren | * Inleveren conceptversie * Verwerken feedback * Drukken verslag * Voorbereiden presentatie | 15 uur | 18/01/2021 |

In onze bronnenlijst valt nog exact na te lezen welke specifieke boeken wij hebben gebruikt en van welke internetsites wij gebruik hebben gemaakt.

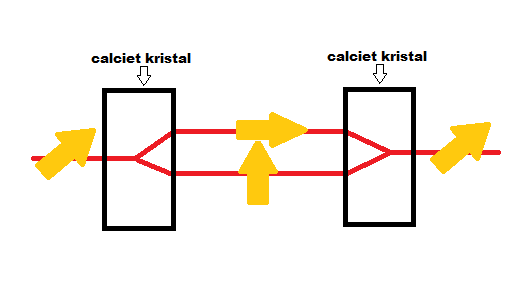
# Hoe werken kwantumcomputers?

Kwantumcomputers zijn enorm snelle machines die heel veel data kunnen analyseren. De naam zegt het eigenlijk al, kwantumcomputers maken gebruik van kwantummechanica. Om te begrijpen hoe zo’n kwantumcomputer nou precies werkt, is het eerst belangrijk om te weten hoe de kwantummechanica in zijn werk gaat.

Figuur 1 Elementair deeltje

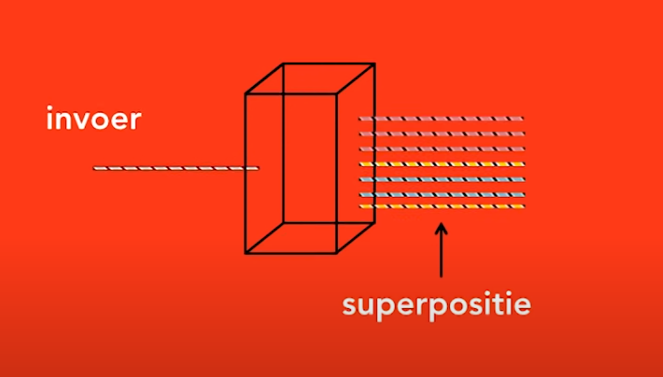
## Kwantummechanica

Kwantummechanica is een deel van de natuurkunde. Het gaat eigenlijk in op de allerkleinste deeltjes van het heelal, elementaire deeltjes. Deze deeltjes kunnen gezien worden als energie pakketjes, zo’n elementair deeltje is namelijk geen golf en geen deeltje, maar het is beiden tegelijk. De deeltjes vertonen dus golfeigenschappen. In figuur 1 valt het beste te zien hoe zo’n deeltje er uit ziet. Volgens de kwantummechanica kunnen deze deeltjes zich tegelijkertijd op verschillende plaatsen bevinden, deze plaatsen noem je toestanden. Een toestand wordt gezien als golf, deze golven kunnen elkaar versterken en verzwakken, door middel van interferentie. Bij interferentie is er sprake van constructieve- (versterken) en destructieve (verzwakken) interferentie. Er ontstaat een waarschijnlijkheidsverdeling, een golfpatroon waarbij een deeltje een grotere kans heeft om op een plaats te bevinden waar de amplitude maximaal is en een kleinere kans heeft om zich op een plaats te bevinden waar de amplitude minimaal is. Hoe meer deeltjes er betrokken zijn, hoe meer verschillende combinaties er mogelijk zijn en daardoor kunnen deze deeltjes zich in nog meer verschillende toestanden bevinden. Dit principe, dat deeltjes zich tegelijkertijd op verschillende plaatsen kunnen bevinden, heet superpositie.

In de kwantummechanica kan er alleen maar gekeken worden naar één moment, het signaal wordt als het ware verstoord en hierdoor is een van de mogelijke toestanden zichtbaar. De ene keer zal een deeltje zich in de ene toestand bevinden en de andere keer in de andere toestand. De kwantummechanica geeft dus eigenlijk aan wat de kans is dat een deeltje zich in een bepaalde toestand bevindt. Prof. dr. Harry Buhrman legde dit enorm goed uit door middel van een voorbeeld met calciet. Calciet breekt invallend licht op in tweeën, ook wel een dubbelbreking genoemd. Licht bestaat uit fotonen, een van de elementaire deeltjes. Deze fotonen hebben allemaal een verschillende richting, ook wel polarisatie genoemd, en deze richting valt te meten, maar het is niet zeker in welke richting dit telkens zal zijn, omdat de fotonen in superpositie zijn. In figuur 2 is het voorbeeld, die prof. dr. Harry Buhrman heeft gebruikt, geschetst. Het ingevallen foton wordt gebroken in een polarisatie naar rechts en naar boven. Bij de tweede calciet vindt interferentie plaats, hierdoor worden deze twee polarisaties weer opgeteld. Doordat er hier maar één foton bij betrokken is, zijn er maar twee mogelijke toestanden. Wat altijd zeker is, is dat het foton een richting heeft, daarom zijn alle toestanden bij elkaar opgeteld 100%. In dit geval zijn er twee mogelijke toestanden, wat dus betekent dat er op elke toestand 50% kans is dat het foton zich hier bevindt. Naarmate het aantal toestanden toeneemt, neemt de kans op elke toestand af, bij vier toestanden is de kans maar 25%. Als al deze toestanden weer interfereren, dan is de kans altijd weer 100%.

Figuur 2 Voorbeeld Prof. Dr. Harry Buhrman

## Kwantumcomputer

Een computer maakt gebruik van bits. Zo’n bit heeft twee waarden, dit zijn de waarden 0 en 1. Door middel van verschillende combinaties van deze nullen en enen, kan er informatie opgeslagen worden en kunnen er berekeningen gemaakt worden. Een kwantumcomputer maakt gebruik van kwantum bits, een bit die in superpositie is van 0 en 1. Naarmate het aantal kwantum bits toeneemt, neemt het aantal toestanden ook toe, er is hier sprake van exponentiële groei met een groeifactor van 2. Doordat een kwantum bit zoveel verschillende toestanden heeft, kunnen er ontzettend veel verschillende berekeningen tegelijkertijd plaatsvinden. Alleen als je het proces verstoort, wordt een van de mogelijke toestanden zichtbaar en krijg je een willekeurig antwoord. Om een juist antwoord te krijgen op de invoer van de kwantumcomputer, is interferentie enorm belangrijk. De waarden 0 en 1 zijn, in de kwantumcomputer, eigenlijk golven. De waarde 0 is een negatieve golf en de waarde 1 is een positieve golf. Door interferentie komen deze golven bij elkaar en versterken ze elkaar of verzwakken ze elkaar. Elke toestand waarop een berekening uitgevoerd kan worden, kan dus een positieve of negatieve toestand hebben. Alle negatieve toestanden vallen eigenlijk weg en worden niet meer meegenomen in de uitvoer, terwijl alle positieve toestanden wel worden meegenomen in de uitvoer. Deze interferentie zorgt ervoor dat het juiste antwoord komt bij de ingevulde invoer. Deze interferentie wordt, net zoals bij de proef van Prof. Dr. Harry Buhrman, met kristallen verkregen.

Figuur 3 kwantum bit

Kwantumdeeltjes zijn deeltjes en golven tegelijk. Om data in te voeren in de kwantumcomputer is een verandering van de staat van een kwantum bit nodig. Hiervoor wordt er gebruik gemaakt van het zo geheten ‘quantum state transfer’. Dit houdt in dat er gebruik gemaakt wordt van een golf met kwantumeigenschappen, om zo de staat van de kwantum bit te veranderen van 0 naar 1 of andersom. Door ‘quantum state transfer’ is het mogelijk om bepaalde informatie op te slaan in het geheugen, maar ook informatie te implementeren in de processor en het kwantumnetwerk.

## Algoritmes

Net zoals bij een normale computer, kunnen kwantumcomputers gebruik maken van algoritmes. Algoritmes zijn een soort van stappenplannen, er is een beginsituatie en een gewilde uitkomst met een aantal stappen om naar de gewilde uitkomst te komen. Doordat de kwantumcomputer niet persé sneller rekent, maar anders, zullen deze algoritmes op een hele andere manier in elkaar steken.

### Shor

Zo ook het algoritme van Shor. Dit is een kwantumalgoritme dat Peter Williston Shor heeft ontwikkeld voor het ontbinden in priemfactoren. Het algoritme kan gebruikt worden om bijvoorbeeld encryptiemethoden die gebruik maken van vermenigvuldiging van priemgetallen, zoals het RSA-systeem, te decoderen. Dit algoritme kan alleen op een kwantumcomputer worden uitgevoerd, hierom spreken we van een kwantumalgoritme. Veenstra heeft dit kwantumalgoritme in haar scriptie verwerkt en uitgelegd. Zij schreef het volgende stappenplan:

“ **Input:**

1. Een geheel getal *n* samengesteld uit twee priemgetallen *p* en *q* waarbij *p* ≠ *q*.

**Algoritme:**

1. Als *n* even is, dan is 2 een factor.

2. Als *n* = *pq* met *p* ≥ 2 en *q* ≥ 2, dan is *p* een factor.

3. Kies een willekeurig getal *x* zodat 1 < *x* ≤ n − 1. Als ggd(*x*, *n*) > 1, dan is ggd(*x*, *n*) een factor.

4. Gebruik het kwantumalgoritme om de orde *r* te bepalen van *x* mod *n*.

5. Als *r* oneven, ga dan terug naar stap 3.

6. Als *r* even maar *x r*/2 ≡ −1 mod *n*, ga dan terug naar stap 3.

7. Anders: bereken ggd(*x**r*/2 − 1, *n*) en ggd(*xr*/2 + 1, *n*). Als de uitkomsten hiervan niet gelijk zijn aan 1 en *n*, dan zijn dit de factoren van modulus *n*.

**Output:**

1. Twee verschillende priemgetallen *p* en *q* zodat *n* = *p* · *q*. “[[1]](#footnote-2)

In dit stappenplan wordt gesproken over ggd, mod, *r* en het kwantumalgoritme. Onder ggd wordt de grootste gemeenschappelijke deler verstaan, dit is het grootste getal waardoor beiden getallen gedeeld kunnen worden. De mod is een afkorting voor modulus, met de modulus wordt telkens de resterende berekend. Er wordt een periode vastgesteld die zich telkens herhaalt, net zoals een klok. Als de klok op 11 uur staat en je bent 6 uur verder, is het niet 17 uur, maar 5 uur. Hier is er dus een periode van 12 uur van toepassing, deze periode wordt er telkens vanaf getrokken en zo houd je de resterende over, in dit geval 5. Er wordt ook van *r* gesproken, dit is de orde. De orde is het kleinste gehele getal *r* waarbij *x r* ≡ 1 mod *n*. Hierbij wordt 0 niet meegeteld, want 40=1, hier komt dus altijd 1(mod 7) uit. Als laatste wordt er gesproken over het kwantum algoritme, dit staat uitgelegd op bladzijde 16 van bijlage 1. Bij stap 2 wordt er gesproken over *p* en *q*, dit wordt bepaald door middel van brute force.

*Voorbeeld 1:*

De orde *r* bepalen van 4(mod 7).

De machten van 4(mod 7):

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Zoals te zien valt is het kleinste gehele getal waarbij *x r* ≡ 1 mod *n,* 43≡ 1 (mod 7), de orde is dus 3.

*Voorbeeld 2:*

**Input:**

1. *n* = *p* · *q*.  *p* is 3 en *q* is 5, dus *n* is 15.

**Algoritme:**

1. *n* is 15, dit is niet even, 2 is dus geen factor
2. Deze stap kan niet gedaan worden, hiervoor moet je brute force uitvoeren op een kwantumcomputer.
3. Het getal *x*, is 12. Ggd(12,15)= 3, dit is groter dan 1, 3 is dus een factor.
4. Het kwantumalgoritme kan niet uitgevoerd worden op een normale computer, de orde wordt op de manier die in voorbeeld 1 uitgelegd is, bepaald. De orde *r* van 12(mod 15) is 3
5. *r* is oneven, dus we gaan terug naar stap 3.
6. Het getal *x*, is 6. Ggd(6,15)=3, dit is groter dan 1, 3 is dus een factor.
7. Het kwantumalgoritme kan niet uitgevoerd worden op een normale computer, de orde wordt op de manier die in voorbeeld 1 uitgelegd is, bepaald. De orde *r* van 6(mod 15) is 2
8. 2 is even
9. *xr*/2 ≡ −1 mod *n*, 66/2= 216 en -1(mod 15) bij 216 is -9 en dus niet -1.
10. ggd(*6**6*/2 − 1, *15*)=5 en ggd(*66*/2 + 1, *15*)= 1, een van de uitkomsten is gelijk aan 1 dus weer terug naar stap 3.

Dit blijft herhaald worden tot er aan stap 7 is voldaan en de priemfactoren 3 en 5 zijn gevonden.

**Output:**

Priemgetallen *3* en *5* zodat *15* = *3* · *5*.

# Wat is de ontwikkeling van (kwantum) computers?

In de afgelopen 100 jaar, zijn er enorme ontwikkelingen geweest omtrent de computer. Eén van de machines die wij, als huidige inwoner in de maatschappij, niet meer kunnen wegdenken uit ons dagelijks leven. Dit was vroeger natuurlijk heel anders.

Voordat er überhaupt computers waren, werd er gebruik gemaakt van rekenmachines. Pas 200 jaar na het bouwen van de eerste rekenmachine, in 1833, kwam Babbage met het idee van de “analytische machine”. Deze machine zou met ponskaarten, oude kartonnen papiertjes met gaatjes erin waarmee informatie kon worden opgeslagen, wiskundige berekeningen kunnen uitvoeren. Deze machine is nooit echt gemaakt en was dus helaas maar een idee. In 1938, kwam Konrad Zuse met de eerste mechanische computer, de Z1. Hij had voor deze computer als eerste het binaire stelsel gebruikt. Enkele jaren later bouwde hij ook de eerste functionele elektronische computer, de Z3.

Door de tweede wereldoorlog, kwamen er steeds meer ontwikkelingen in de computerwereld. De Duitse berichten die gecodeerd werden door de Enigma-machine en de Lorentz-machine, moesten worden onderschept en gedecodeerd worden. Er werd hiervoor een geheime operatie in stand gezet. Max Newman en Tommy Flowers ontwikkelden de eerste elektronische computer, de Colossus Mark 1. Deze werd in gebruik genomen in Engeland in januari 1944. Na een aantal maanden, juni 1944, volgde ook de verbeterde versie de Colossus Mark 2.

De eerste computer, de ARRA 1, kwam vrij snel na de tweede wereldoorlog in Nederland, dit was al in 1952, en deze was ontwikkeld door de Universiteit van Amsterdam. Rond de jaren ’70 werd er gebouwd aan een c0mputer die ook thuis te gebruiken was, dus niet alleen op het werk. In augustus 1981 werd door IBM deze computer gepresenteerd aan het publiek, dit was de eerste personal computer (PC). Doordat de ontwikkelaars van deze computer alleen maar onderdelen gebruikten die op de markt te koop waren en zelf een boek verkochten om een computer te bouwen, kwamen er veel dezelfde computers op de markt. Dit zorgde voor een grote concurrentie en zo ontwikkelde de PC zich enorm rap, dit gebeurt tot op de dag van vandaag nog steeds.

Wat erg verbazingwekkend is, is dat al in 1980 het idee van de kwantumcomputer ontstond. Dit idee werd een jaar later gepresenteerd op een conferentie van de MIT en IBM, het bedrijf dat de PC ontwikkelde. In de 30 jaar hierna werden kwantum protocollen en de nodige voorwaarden voor het bouwen van zo’n kwantumcomputer gepubliceerd. Daarbij werden ook verschillende uitvindingen gedaan zoals de semiconductor chip ion trap en werd de eerste kwantum processor gebouwd. Pas vanaf 2011, toen er een belangrijke doorbraak kwam, nam het ontwikkelen van de kwantumcomputer een snelle vaart. Nog geen vier jaar later werd de eerste kwantumcomputer gebouwd door NASA en D-Wave Systems. Het jaar hierop maakte IBM de krachtigste kwantumcomputer met een processor van 16 kwantum bits.

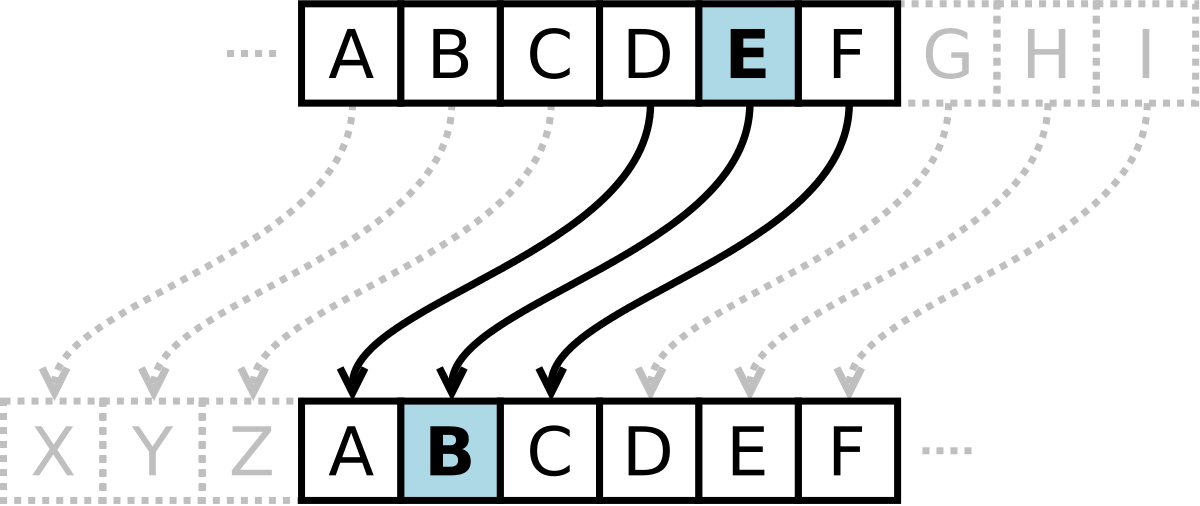
In de afgelopen 40 jaar heeft de kwantumcomputer zich ontwikkeld van een idee tot een in het echt bestaande machine. Het ontwikkelen van de kwantumcomputer is nog lang niet klaar, men is pas bij het begin. Een voorbeeld hiervan is ook de ‘normale’ computer. Sinds de eerste computer geïntroduceerd werd, bleef de computer steeds beter worden. De computer is in de afgelopen jaren steeds kleiner, gebruiksvriendelijker en sneller geworden. Een grote kans is dat dit ook zal gebeuren met kwantumcomputers. Alleen commerciële toepassingen vormen nog een grote uitdaging. Onder invloed van de omgeving kunnen kwantumtoestanden zoals verstrengeling en superpositie namelijk gemakkelijk verloren gaan. Naar een goede bescherming van de kwantumtoestand wordt dan ook hard gezocht. Dit zal niet eenvoudig zijn, maar de kans is er dat zo’n kwantumcomputer, die nu in verschillende laboratoriums gebruikt en verbeterd wordt, uiteindelijk bij ons thuis op het bureau staat.

# Hoe is het RSA-systeem tot stand gekomen?

Cryptografie is een aftakking in de wiskunde waarbij de versleuteling van boodschappen wordt bestudeerd. De cryptografie is in de tijd van Julius Caesar voor het eerst gebruikt en wordt hedendaags ook in de kwantumtechnologie gebruikt. De methode die Julius Caesar heeft bedacht, de Caesar Cipher, was het begin van de opkomst van het RSA-systeem.

## Caesar-Cipher

Caesar-Cipher is een van de oudste en bekendste geheimschriften. Dit geheimschrift is genoemd door Julius Caesar, die deze methode gebruikte om boodschappen naar zijn leger te sturen, met de zekerheid dat deze boodschap niet begrepen kan worden door de vijanden. Het Caesar-Cipher is heel eenvoudig, Caesar codeerde zijn berichten door A te vervangen door D en B door E enzovoort, dus elke letter heeft de betekenis van de letter n (aantal) plaatsen ervoor in het alfabet, of juist verderop. Een bekende uitspraak van Caesar was YHQL, YLGL, YLFL wat staat voor veni, vidi, vici, wat weer “ik kwam, ik zag, ik overwon” betekent. Hierbij heeft n de waarde van 3.

Als dit geheimschrift in een functievoorschrift geschreven moet worden gaat dat als volgt, alle letters van het alfabet krijgen een getal: A=0, B=1, C=2 enzovoort. Aangezien elke letter met de letter 3 plaatsen van rechts wordt geschreven wordt het functievoorschrift: f(x)= x+3. Bij de letters X, Y en Z klopt deze formule niet. Dit komt omdat de letters aan het eind van het alfabet staan en als de rangnummers van deze letters ingevuld worden in de formule, namelijk x= 23, 24 of 25, de uitkomst niet zal overeenkomen met het de getallen 0,1 en 2, die voor A, B en C staan. Hiervoor moet modulus 25 gebruikt worden. Hierbij worden alleen gehele getallen gebruikt groter dan 25. Dus als de uitkomst van f(x)=x+3 groter is dan 25 wordt 25 van de uitkomst getrokken totdat de uitkomst daarvan kleiner is dan 25.

*Voorbeeld 3*:

F(25)=28

28>25

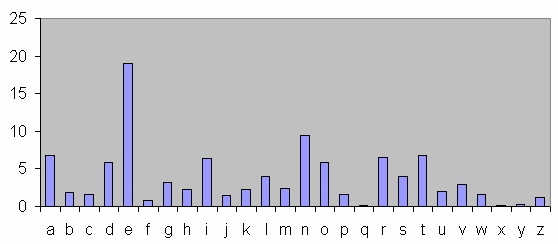
28(mod25)=

28-25= 3

3= letter C

Figuur 4 Caesar-Cipher

800 jaar later werd de zwakte van de caesarversleuteling bekend gemaakt. Het werd ontdekt door de Arabische wiskunde Al-Kindi. Hij heeft met behulp van frequentieanalyse het geheim achter het Caesar Cipher ontdekt. Voor elke taal geldt dat een letter in een tekst meer voorkomt dan een ander, dit bepaalt de frequentie van deze letter in de taal zelf. Om een overzicht te maken van de frequentie van de meest gebruikte letters, gaat men een analyse maken van een boek in een bepaalde taal bijvoorbeeld het Engels, hiermee gaat hij een frequentietabel maken waarin duidelijk is welke letter het meest voorkomt in deze taal.

 Om een gecodeerde tekst te decoderen, telt men de frequentie van elke letter in de gecodeerde tekst en zoekt hoever het van de oorspronkelijke letter is verschoven. Stel dat de communicatie in het Engels is en de meest voorkomende letter in een gecodeerde tekst is H, dit betekent dat de H hoogstwaarschijnlijk voor de E staat, 3 plaatsen ervoor, omdat de E de meest voorkomende letter is in het Engels (zie figuur 5). Zo kan de afluisteraar de code van de tekst verder breken door alle gecodeerde letters 3 plaatsen naar achter te schuiven in het alfabet. Toen Al-Kindi de zwakte van het Caesar Cipher heeft bekend gemaakt, kwam er een einde aan de veiligheid van deze encryptiemethode.

Figuur 5: Engelse letters-frequentieoverzicht

## Atbash-geheimschrift

Een ander geheimschrift, dat een joodse oorsprong heeft, is het Atbash-geheimschrift. Hierbij wordt de letter A geschreven als Z, B als Y en C als X enzovoort. Om een functievoorschrift hiervan te kunnen maken, krijgen alle letters weer een getal zoals bij het geheimschrift Caesar-Cipher. Het functievoorschrift van dit geheimschrift wordt f(x)=25-x. Het getal 25 staat voor het aantal letters in het alfabet, van A=0 tot Z=25.

## Blaise de Vigenère

Omdat de Caesar-Cipher te eenvoudig te kraken werd, heeft de Franse diplomaat Blaise de Vigenère in de 16e eeuw een nieuwe encryptiemethode ontwikkeld. Hij heeft na zijn pensionering een boek geschreven, waarin hij een nieuwe manier van versleuteling beschreef. Zijn manier was om elke letter steeds door een andere, niet vaste letter te vervangen. Dit kan gerealiseerd worden door een codewoord elk rangnummer van een letter van het codewoord bij elk rangnummer van een letter van de boodschap op te tellen. Elke letter heeft een nummer zoals boven beschreven, A=00, B=01, Q=16, enzovoort. Door de nummers van de letters van het codewoord en de nummers van de letters van de boodschap op te tellen, kan de Vigenère-Cipher gemaakt worden. Het codewoord wordt hierbij telkens herhaald tot aan de laatste letter van de boodschap.

*Voorbeeld 4*:

Als we het codewoord *nagel* nemen en we willen een boodschap versturen. De boodschap is: *ik houd van eten*. Dan kunnen we de volgende tabel opstellen en hieruit de code halen.

Codewoord: nagel

Boodschap: ik houd van eten

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Boodschap | I  08 | K  10 | H  07 | O  14 | U  20 | D  03 | V  21 | A  00 | N  13 | E  04 | T  19 | E  04 | N  13 |
| Codewoord | N  13 | A  00 | G  06 | E  04 | L  11 | N  13 | A  00 | G  06 | E  04 | L  11 | N  13 | A  00 | G  06 |
| Uitkomst getal | 21 | 10 | 13 | 18 | 31 | 16 | 21 | 06 | 17 | 15 | 32 | 04 | 19 |
| Code | V | K | N | S | F | Q | V | G | R | P | G | E | T |

Code: VKNSFQVGRPGET

Wanneer de uitkomst van de som, van de getallen van het codewoord en de boodschap, groter is dan 26 dan geldt: uitkomst-26= getal van de codeletter.

## Het vernam systeem

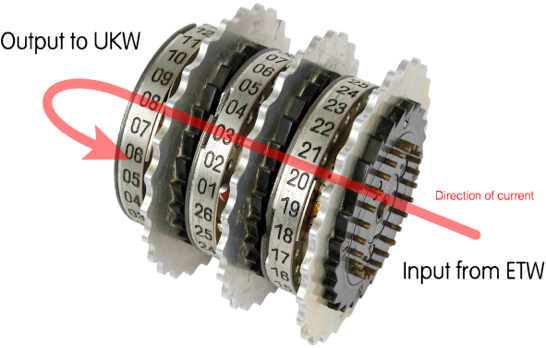
In de loop van de 20e eeuw heeft de Amerikaan Gilbert Vernam een methode bedacht die voor een tijdje moeilijk te kraken was door de Duitsers. De tekst die gecodeerd moest worden, moest eerst versleuteld worden met een sleutel. Hierbij was er ook sprake van een codewoord, net zoals bij de methode van Vigenère. Het enige verschil is dat het codewoord hierbij langer is. Maar zodra het codewoord bekend is door de kraker is deze methode niet meer veilig.

Om ervoor te zorgen dat de methode veilig blijft, zijn er een aantal voorwaarden:

* Persoon A maakt een bericht, x, en versleutelt die met zijn lange codewoord (A) en stuurt die in de vorm van A(x) op naar persoon B.
* Persoon B vercijfert A(x) met zijn eigen codewoord B(x) en stuurt hij die op in de vorm van B(A(x)) naar persoon A.
* Persoon A haalt zijn eigen code van B(A(x)) weer weg om B(x) over te houden en die vervolgens naar persoon B te sturen.
* Persoon B moet zijn code van B(x) afhalen om het bericht te kunnen begrijpen.

Op deze manier kan het probleem wel opgelost worden, want er hoeven geen codes uitgewisseld te worden. Maar helemaal veilig is het nog lang niet, omdat de kans dat de code bij het derde bericht vertaald wordt relatief groot is en hierdoor kan de kraker de eerder gestuurde berichten ook ontcijferen

## Enigma-code

 De Enigma-code werd veel gebruikt tijdens de Tweede Wereldoorlog. Het werd gebruikt door de Duitsers en de techniek was al ontworpen door Hebern in 1920. In Amerika werd er gebruikgemaakt van soortgelijke methode genaamd Sigaba en in Japan was dat Red en Purple. Deze machines zijn rotorsystemen. Een rotor is een schijf in isolerend materiaal, waar aan beide kanten elektrische contactpunten zitten, een voor elk symbool van het gebruikte alfabet. Aan de binnenkant is elk contactpunt aan de ene kant verbonden met het contactpunt van de andere kant. Hierdoor ontstaat permutatiecode (gehusselde Caesar code). De machines hadden drie of vier van zulke rotoren, waarvan de posities bij elke gecodeerde letter wijzigden.

Aan het begin werd er gebruik gemaakt van een codeboek om de begininstelling van de Enigma te kiezen, deze begininstelling was elke dag weer anders. Na elke omwenteling draait de tweede en de derde rotor een plaats door. Een voorbeeld is dat de begininstelling ervoor zorgde dat de ingetypte C veranderde in K en de tweede rotor veranderde de K in R en de derde veranderde die in U. Dit betekent dat de C in U is veranderd. Omdat de laatste 3 rotoren doordraaiden waren er 263= 17576 posities van de rotoren mogelijk. Men dacht dat het decoderen van deze methode moeilijk was, maar tot hun verbazing, werd de Enigma-code toch gekraakt door de Engelse wiskundige Alan Turing.

## Data encryption standard

De Data Encryption Standard (DES) methode is in de jaren ‘70 ontwikkeld door IBM voor gebruik door de overheid en privésector. Dit systeem werkt volgens een blok-codering. Hierbij wordt er gebruik gemaakt van blokken van 64 bits, die tegelijk worden gecodeerd door een computerchip. Een blok bevat 56 bits en een controle van 8 controle bits. DES is gebaseerd op een encryptiemethode dat Lucifer Cipher wordt genoemd. De werking van de DES gaat als volgt:

* Een blok van 64 bits bestaat uit 64 getallen.
* Met behulp van permutatie worden de 8 controle bits uit de reeks getallen gehaald en worden de 56 over gebleven getallen door elkaar gehusseld.
* Hierna worden de 56 bits in 2 helften gesplitst, elke helft bestaat uit 28 getallen.
* Elke helft ondergaat 16 rondes van het encryptieproces
* Wanneer deze 16 rondes voltooit zijn, komen de twee helften samen in de uiteindelijke, nieuwe gevormde blok met 16 bits.

Door de 56 bits waren er 256 sleutels mogelijk en werd het daarom eerst gezien als een onmogelijk te kraken methode, maar daar is verandering in gebracht na het decoderen van de sleutel binnen 96 dagen, door gebruik te maken van computers.

In 2001 werd deze methode verder ontwikkeld tot het Advanced Encryption Standard (AES) waarbij er gebruik gemaakt wordt van blokken van 128 of zelfs 256 bits waarbij het kraken wel moeilijker werd.

## Diffie-Hellman

In 1976 werd er een nieuwe methode ontwikkeld door Whitefield Diffie en Martin Hellman. Bij deze methode werd het eindelijk mogelijk om een decodeersleutel met iemand te delen zonder moeilijk te doen om de sleutel op een veilige manier bij de ontvanger te krijgen. Hoe deze methode werkt wordt bij de volgende deelvraag uitgelegd omdat de Diffie-Hellman sleuteluitwisseling een grote invloed heeft op het bestaan van het RSA-systeem.

## Het RSA-systeem

De Diffie-Hellman sleuteluitwisseling heeft tot het ontstaan van het RSA-systeem geleid. Het RSA is in 1978 ontwikkeld en het staat voor de bedenkers van het systeem namelijk, Ron **R**ivest, Adi **S**hamir en Leonard **A**dleman. Deze twee informatici en een wiskundige aan het MIT in Boston vonden het interessant hoe de Diffie-Hellman methode werkte en hebben er daarom verbetering in aangebracht.

Het RSA is tegenwoordig een van de meest gebruikte coderingssystemen. Het RSA- systeem dient voor de beveiliging van de informatie en het biedt ook de mogelijkheden om de echtheid van berichten te garanderen. Hoe het RSA-systeem werkt wordt in de volgende deelvraag besproken.

# Hoe werkt het RSA-systeem?

Cryptografie, ofwel geheimschrift, bestaat sinds de tijd van de Romeinen. De Romeinen gebruikten geheimschrift in bijvoorbeeld het leger. Cryptografie is al eeuwenlang een wedstrijd tussen code makers en code brekers, omdat code brekers het geheimschrift steeds wisten te kraken, moesten code makers steeds nieuwe manieren vinden om de boodschappen geheim te houden. Cryptografie heeft een grote ontwikkeling door gemaakt tijdens de Tweede Wereldoorlog. Tijdens de Tweede Wereldoorlog werd er daarom gebruik gemaakt van een geheimschriftmachine door de Duitsers om hun onderlinge communicatie te beveiligen. Tot hun verbazing, wisten de Engelsen toch deze machine te kraken waardoor ze van alle communicaties en plannen van de Duitsers op de hoogte waren.

Door de ontwikkeling van de computer werd voor de code brekers het kraken van geheimschrift met weinig moeite gedaan. Maar in de loop van de tijd heeft cryptografie zich ook weten te ontwikkelen om ervoor te zorgen dat ze niet te eenvoudig te decoderen zijn door de computer. Men heeft hiervoor het RSA-systeem bedacht.

## Geschiedenis van het RSA-systeem

De Diffie en Hellman methode is in de jaren 60 ontdekt door de Amerikaanse cryptograaf Whitefield Diffie die het belang van contact via het internet zag. Niet alleen het contact van de regering via het internet was van belang, maar ook die van de gewone mens. Hij vond dat ieder persoon recht had op privacy, ook op internet. Om aan deze beveiliging te kunnen voldoen, had hij een encryptiemethode uitgewerkt waarbij er geen sleutel gewisseld hoefde te worden. Deze methode had Diffie samen met Hellman uitgewerkt tot dat het mogelijk werd om een decodeersleutel met iemand te delen zonder de moeite te doen om de boodschap op een veilige manier bij de ander te krijgen. Deze Diffie-Hellman sleuteluitwisseling werd in 1976 ingevoerd.

Het doel van de Diffie-Hellman methode is het voorkomen dat de boodschap van de zender aan de ontvanger gekraakt wordt door de afluisteraar. Hiervoor moeten de zender en de ontvanger een gezamenlijke sleutel hebben die de afluisteraar niet kan onderscheppen. Bij de Diffie en Hellman encryptiemethode werd er gebruik gemaakt van de eenrichtingsfunctie. Bij deze functie is het gemakkelijk om die functie toe te passen, maar aan de hand van de uitkomst is het lastig om de ingevulde waarde te achterhalen. Voor deze encryptiemethode was een contactmoment tussen de zender en ontvanger wel van belang om een sleutel af te spreken, zonder de sleutel zelf te overhandigen.

Dit contactmoment werd als onhandig gezien door de drie cryptografen Rivest, Shamir en Adleman. Ze hebben daarom de encryptiemethode van Diffie en Hellman verbeterd. Bij de verbeterde versie die deze 3 cryptografen bedacht hebben was het niet meer nodig om een sleutel af te spreken tussen de zender en de ontvanger. Wel was het belangrijk dat er openbare en geheime sleutels waren, maar bij het RSA-systeem hoeven die sleutels alleen van de ontvanger te zijn. De ontvanger publiceert de openbare sleutel waarmee anderen berichten aan de ontvanger kunnen coderen. De verzender zoekt deze openbare sleutel op en gebruikt deze om het te verzenden bericht te versleutelen. Vervolgens gebruikt de ontvanger zijn geheime sleutel om het ontvangen (versleutelde) bericht te kunnen decoderen.

## Werking van het RSA-systeem

Om de werking van het RSA-systeem gedetailleerd te begrijpen is het van belang om de volgende elementen te kennen.

### Priemgetallen

Een priemgetal is een getal dat groter is dan 1 en alleen te delen is door 1 en door zichzelf. Een priemgetal is dus een 2-deler. Om erachter te komen of een getal een priemgetal is, kan de volgende methode gevolgd worden:

Eerst wordt het getal gedeeld door alle priemgetallen die tussen 1 en n (het getal zelf) liggen. Als de uitkomst een natuurlijk getal is zonder decimalen en breuken dan is het getal een samengesteld getal en dus niet priem.

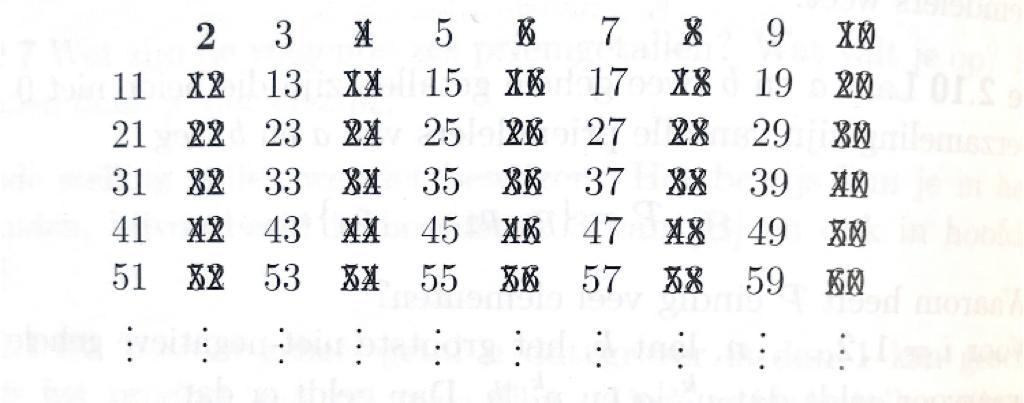
De eerste 10 priemgetallen zijn: 2, 3, 5, 7, 11, 13, 17, 19, 23, 29

Wat ook belangrijk is om te weten in de getaltheorie is dat elk geheel positief getal n dat groter is dan 1 geschreven kan worden als een product van een aantal priemgetallen.

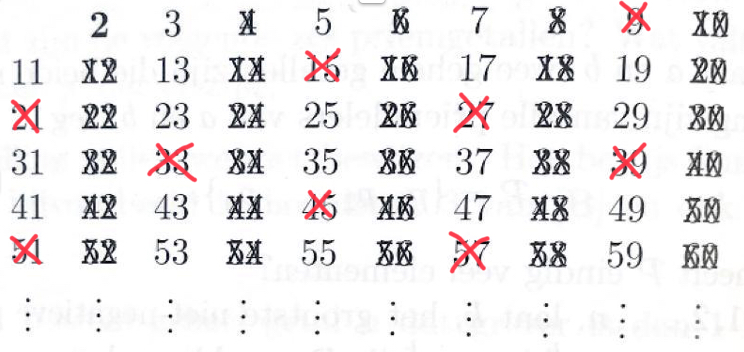
*Voorbeeld 5:*

Het getal 156

156= 2 x 2 x 3 x 13

Het vinden van priemgetallen wordt steeds lastiger naarmate de getallen groter worden. Eratosthenes heeft hiervoor een manier gevonden waarbij de priemgetallen eenvoudig gezeefd kunnen worden. Zijn idee gaat als volgt:

Alle positieve natuurlijke getallen die veelvoud zijn van het priemfactor 2 zijn geen priemgetallen. Dus 4, 6, 8, 10, …, 2n. Deze getallen kunnen er weg gestreept worden (zie de afbeelding hiernaast)

Dit gebeurt ook met het volgende priemgetal namelijk 3. Alle positieve natuurlijke getallen die veelvoud zijn van 3, zoals 6, 9, 12, ..., 3n kunnen worden weggestreept.

Dit gaat daarna door naar het volgende priemgetal, wat 5 is en daarna 7 enzovoort. Het getal dat niet weggestreept is, zal het volgende priemgetal zijn.

### Algoritme van Euclides

Bij het algoritme van Euclides is het van belang om te weten wat ggd betekent. Ggd staat voor de grootste gemene deler tussen 2 factoren.

*Voorbeeld 6:*

Ggd(12,36)= 12

Naarmate de getallen steeds groter worden, wordt het ook steeds lastiger om de grootste gemene deler te vinden en te berekenen. Hiervoor is het algoritme van Euclides bedacht dat nog steeds up-to-date is.

Voor dit algoritme geldt: ggd(a,b). Beide getallen zijn positief en gehele getallen. Neem aan dat a > b. Hierbij past b zoveel keer in a en er blijft ook een rest *r1* over. Voor r1 geldt: 0 < r1 < b. Hierbij ontstaat er een positieve geheel getal k1 en een niet-negatief getal r1. Wanneer r­1 = 0, dan is de oplossing gevonden.

Dus: a= k1b+ r1 met 0 < r1 < b

Uit deze formule kan er verklaard worden dat als d een deler is van a en b, dan is d ook een deler van b en r1 en omgekeerd. Dus als d een deler is van b en r1, dan is het ook een deler van a.

Er geldt:

Ggd(a,b) = ggd(b,r1)

Wanneer r1= 0, dus er is geen rest, dan geldt dat ggd(a,b) = ggd(b,0) = b

Als r1 niet gelijk is aan 0, dan ontstaat een positief geheel getal k2 en een niet-negatief getal r2.

De formule wordt: b = k2r1 + r2 met 0 < r2 < r1

En wanneer r2= 0 dan geldt dat ggd(a,b)=r1. Als r2 niet gelijk is aan 0, dan ontstaat een positief geheeld getal k3 en een niet-negatief getal r3.

Er geldt dus: r1 = k3r2 + r3

Dit blijft doorgaan totdat rn > 0 en rn+1 = 0

Hierbij ontstaat het volgende:

Ggd(a,b) = ggd(b, r1) = ggd(r1, r2) = ggd(r2, r3) = … = ggd(rn, rn+1) = ggd(rn, 0) = rn

Dus: ggd(a, b) = rn

*Voorbeeld 7:*

a= 120

b= 84

k1= 1

r1= 36

a= k1b+ r1

120 = 1\*84 + 36

0< r1< b

Dus:

b = k2r1 + r2

84 = 2\*36 + 12

0< r2 < r1

Dus

r1 = k3r2 + r3

36 = 3\*12 + 0

r3 = 0

dus ggd(a,b) = r2

dus ggd(120, 84) = 12

*Voorbeeld 8a:*

Toon aan dat ggd(a, b) = 3 met a= 654321 en b=123456

654321 = 5\*123456 + 37041

123456 = 3\*37041 + 12333

37041 = 3\*12333 + 42

12333 = 293\*42 + 27

42 = 1\*27 + 15

27 = 1\*15 + 12

15 = 1\*12 + 3

12 = 4\*3 + 0

Rn+1 = 0

Dus ggd(a,b) = rn

Ggd(654321,123456) = 3

Om de werking van het RSA-systeem met het algoritme van Euclides te begrijpen is het echter ook van belang om de volgende formule te kunnen gebruiken: ax + by = ggd(a,b)

Om deze formule te begrijpen wordt voorbeeld 8a gebruikt. Hierbij worden x en y berekend.

*Voorbeeld 8b:*

654321**x** + 123456**y** = 3

3 = 15 - 12 (15 = 1\*12 + 3)

12= 27 – 1\*15 (27 = 1\*15 + 12)

12 substitueren in 3 = 15 - 12 geeft:

3 = 15 – (27 – 1\*15)

Haakjes wegwerken geeft hetzelfde als:

3 = 15 – 27 + 1\*15

= 2\*15 – 27

15 = 42 – 1\*27

3 = 2\*(42-1\*27) -27

= 2\*42 – 2\*27 – 27

= 2\*42 – 3\*27

27 = 12333 - 293\*42

3 = 2\*42 – 3\*(12333 - 293\*42­)

= 2\*42 - 3\*12333 + 3\*293\*42

= 881\*42 – 3\*12333

42 = 37041 - 3\*12333

3 = 881\*(37041 - 3\*12333) – 3\*12333

= 881\*37041 – 2643\*12333 – 3\*12333

= 881\*37041 – 2646\*12333

12333 = 123456 - 3\*37041

3 = 881\*37041 – 2646\*(123456 - 3\*37041)

= 881\*37041 – 2646\*123456 + 7938\*37041

= 8819\*37041 – 2646\*123456

37041 = 654321 - 5\*123456

3 = 8819\*(654321 - 5\*123456) – 2646\*123456

= 8819\*654321 – 44095\*123456 - 2646\*123456

= 8819\*654321 – 46741\*123456

Hierin is de formule ax + by = ggd(a,b) te herkennen

Ggd(a,b) = 3

A= 654321

B= 123456

X= 8819

Y= -46741

### Modulo rekenen

De term modulus met de afkorting mod is eerder genoemd bij de deelvraag ‘Hoe werken kwantumcomputers?’, onder het kopje ‘Algoritmes’. Modulo rekenen wordt onbewust dagelijks gebruikt door de mensheid, bijvoorbeeld met klokkijken. Men weet dat als het nu 22:00 uur is, is het over 5 uur 3:00 uur. In de wiskunde wordt een veelvoud van een getal dat na het maximum (mod n) komt, congruent genoemd. Dit wordt als volgt geformuleerd: 22 + 5 ☰ 3 (mod 24). Mod 24 staat voor 24 uur, dus het maximum.

De definitie van modulo rekenen is op deze manier te verklaren: voor gehele getallen a en b en een geheel getal n> 1 wordt er gezegd dat a congruent is aan b modulo n. Dit wordt als volgt genoteerd: a ☰ b (mod n)

*Voorbeeld 9:*

Vraag:

a= 30 en b = 57

Wat weet je van a\*b modulo 24?

Oplossing:

Het uiteindelijke antwoord moet als volgt genoteerd worden. a\*b ☰ rest (mod 24)

a\*b = 30\*57 = 1710

1710/24 = 71,25

24\*71= 1704

1710 – 1704= 6 (rest)

Dit wordt geformuleerd als: a = b + kn

Hier wordt a beschouwt als de uitkomst van a\*b, b als rest, k als een geheel getal dat de uitkomst is van a/n en n is de modulus

Dus:

a= 30\*57 = 1710

k= 71

b= 1710 – (71\*24) = 6

n= 24

a = b + kn

1710 = 6 + 71\*24

Notatie:

a\*b ☰ 6 (mod 24)

Wat ook belangrijk is om te weten bij modulo rekenen is het volgende:

Er geldt: a ☰ b (mod n) en b ☰ c (mod n) dan is a ☰ c (mod n)

Dit kan zo bewezen worden, a = b + kn en b = c + hn dan is a = (c + hn) + kn = c + (h + k)n. En omdat h + k een geheel getal is, is c plus een geheel getal van veelvouden van n, gelijk aan a, oftewel a ☰ c (mod n). Dit geldt ook voor getallen met machten.

Als laatst geldt het volgende: a ☰ b (mod n) en c ☰ d (mod n) dan is ac ☰ bd (mod n)

Verklaring: a = b + kn

c = d + ln

ac = (b + kn) (d + ln) = bd + bln + knd + knln = bd + (bl + kd + knl) n

en omdat (bl + kd + knl) een geheel getal is, is ac ☰ bd (mod n)

### Priemfactoratie met Fermat

Pierre de Fermat heeft een grote bijdrage geleverd aan de werking van het RSA- systeem. Hij heeft een methode bedacht waarin men eenvoudig een getal kan factoriseren. Factoriseren betekent het getal schrijven als een product van priemgetallen (zie voorbeeld 5).

De methode van Fermat dat het factoriseren van een getal vergemakkelijkt gaat als volgt.

* Als het getal even is, dan wordt het door 2 gedeeld net zolang dat het oneven wordt.
* Wanneer het getal oneven wordt, dan geldt: n = a \* b waarbij a > b > 1. a en b zijn hierbij ook oneven getallen. Dit komt omdat het product van twee even getallen en het product van een even en een oneven getal, altijd een even uitkomst heeft.
* Nu worden x en y gedefinieerd. x= (a + b)/2 en y= (a – b)/2. x en y zijn hierbij gehele getallen en niet breuken omdat a en b allebei oneven zijn. En de som en verschil bij oneven getallen heeft altijd een even getal.
* Er geldt nu ook dat n = x2 – y2 met x > y > 0

Verklaring: a = x + y want er geldt, x + y = (a + b)/2 + (a – b)/2 = (a + b + a – b)/2 = 2a/a = a

b = x – y want er geldt, x – y = (a + b)/2 - (a – b)/2 = (a + b - a – b)/2 = 2b/2 = b

n = a \* b = (x + y)(x – y) = x2 – y2

* Vervolgens wordt het volgende gedaan. n = x2 – y2 => y2 = x2 – n

Hierbij hoeft er maar een variabele als eerst gebruikt te worden, namelijk de x2 – n. Wanneer de wortel van de uitkomst een geheel getal is, dan is y opgelost.

*Voorbeeld 10:*

n = 2027651281

x= √n + 1

en

y = √x2– n

Wanneer de uitkomst van de wortel niet een geheel getal is dan wordt dit symbool gebruikt †, hierbij moet er verder gezocht worden dus √n + 1 + 1, √n + 1 + 1 + 1, enzovoort tot dat de uitkomst y een geheel getal is.

dus:

√n = 45029,44904

Hierbij wordt 45029 genomen en 1 erbij opgeteld voor x.

(45030)2 – n = 49619 † want √49619 is geen geheel getal.

(45031)2 – n = 139680 †

(45032)2 – n = 229743 †

(45033)2 – n = 319808 †

(45034)2 – n = 409808 †

(45035)2 – n = 499944 †

(45036)2 – n = 590015 †

(45037)2 – n = 680088 †

(45038)2 – n = 770163 †

(45039)2 – n = 860240 †

(45040)2 – n = 950319 †

(45041)2 – n = 1040400

y = √1040400 = 1020

x = 45041

Hierdoor is a en b te berekenen, want n = a \* b

a = x + y en b = x – y

Dit geeft weer a = 45041 + 1020 = 46061 en b = 45041 – 1020 = 44021

Dit zijn priemgetallen die niet eenvoudig berekend kunnen worden. Vandaar dat men vindt dat het begrijpen van de Fermat methode essentieel is bij de werking van het RSA-systeem.

### Diffie en Hellman

Deze term is eerder aan bod gekomen in de vorige deelvraag. Diffie en Hellman hebben een grote bijdrage geleverd aan de opkomst van het RSA-systeem. Dit methode heeft ervoor gezorgd dat het mogelijk werd om een decodeersleutel met iemand te delen zonder moeilijk te hoeven doen om de sleutel op een veilige wijze bij de ander te krijgen.

Bij de Diffie-Hellman sleuteluitwisseling zijn er 3 personen betrokken: de zender, de ontvanger en de afluisteraar. De zender wil de ontvanger een boodschap sturen en wil voorkomen dat de afluisteraar de boodschap onderschept. Hiervoor moet de boodschap gecodeerd worden en er is daarom een codesleutel nodig. Dit gaat als volgt:

* De zender en de ontvanger maken de sleutel. Hierbij wordt er gebruik gemaakt van modulo rekenen. Ze kiezen voor de functie yx (mod p). Ze spreken samen een waarde af van y en p. De waarde moet een geheel getal zijn en p is een priemgetal. p mag niet gelijk zijn aan y.
* De zender en de ontvanger kiezen elk een eigen geheel getal.
* Elk, de zender en de ontvanger, vult zijn eigen gekozen getal in als x-variabele en stuurt zijn uitkomst naar de ander.
* De ontvanger en de zender gebruiken de functie xq(mod p). Hierbij is x de uitkomst van de ander en q het eigen gekozen getal bij stap 2.
* Door dit allemaal in te vullen krijgen de zender en de ontvanger samen een gezamenlijke sleutel.

*Voorbeeld 11:*

p = 7 (priemgetal)

y = 5

Zender eigen gekozen getal = 4

Ontvanger eigen gekozen getal = 2

Zender doet het volgende: 54 (mod 7) ☰ 625 (mod 7) ☰ 2 (mod 7)

Ontvanger doet het volgende: 52 (mod 7) ☰ 25 (mod 7) ☰ 4 (mod 7)

De zender stuurt de uitkomst, 2, naar de ontvanger en de ontvanger doet hetzelfde. Hij stuurt 4 naar de zender.

De zender doet het volgende: x4 (mod 7) met x als 4. Dit wordt:

44 (mod 7) ☰ 256 (mod 7) ☰ 4 (mod 7)

De ontvanger doet x2 (mod 7) met x als 2. Dit wordt:

22 (mod 7) ☰ 4 (mod 7)

De gezamenlijke sleutel is dus 4.

Aan de hand van alleen de uitkomsten die er onderling uitgewisseld worden tussen de zender en de ontvanger is het lastig om door de afluisteraar te bepalen wat de gekozen getallen zijn door beide partijen en hierdoor is het vinden van de gezamenlijke sleutel ook lastig. Deze functie wordt daarom een eenrichtingsfunctie genoemd. Hierbij wordt er bedoeld dat het gemakkelijk is om die functie toe te passen, maar aan de hand van de uitkomst is het lastig om de ingevulde waarde te achterhalen

### Fermat en Euler

Fermat is een van de meest invloedrijke wiskundigen geweest, hoewel hij zelf geen wiskundige was, maar een jurist. Hij correspondeerde met veel wiskundigen uit zijn tijd, maar gaf daarbij bijna altijd het gevonden resultaat zonder het bewijs, die laat hij weg. De stelling van Fermat die met de werking van het RSA-systeem te maken heeft is het volgende: laat p een priemgetal zijn en a een positief geheel getal, zo dat p geen deler is van a. Dan geldt dat a­p-1 ☰ 1 (mod p).

Het bewijs voor deze stelling heeft Fermat niet gepubliceerd. Leonard Euler heeft in 1736 het bewijs van deze stelling achterhaald. Deze stelling werkt als volgt:

* Stel eerst alle p-1 veelvouden van a. Dit zijn a, 2a, 3a, 4a, …, (p-1)a. (Aangezien a niet deelbaar is door p, zijn de veelvouden dus ook niet deelbaar door p.)
* Neem 2 van deze veelvouden en noem ze ra en sa. Hierbij liggen r en s tussen 1 en p – 1. De stelling wordt: ra ☰ sa (mod p). Oftewel p deelt ra – sa want ra – sa ☰ 0 (mod p) of r – s omdat ra – sa = a(r – s) en p is geen deler van a.
* Omdat r en s beide tussen 1 en p – 1 liggen geldt dat r – s tussen –(p – 2) en p – 2 ligt.

*Voorbeeld 12:*

a= 3 en p= 11

* Veelvouden van a tot p – 1

a, 2a, 3a, 4a, 5a, 6a, 7a, 8a, 9a, 10a

3, 6, 9, 12, 15, 18, 21, 24, 27, 30

* Twee veelvouden nemen, bijvoorbeeld 4a en 5a

4a ☰ 5a (mod 11)

* r – s= 4 – 5 = -1

-(p – 2)= -9

p – 2 = 9

r – s ligt dus tussen -9 en 9

* Behalve 0, zijn deze getallen –(p – 2) tot p – 2, allemaal niet deelbaar door p, er moet dus gelden r – s = 0 oftewel r = s. Hieruit valt te concluderen dat a, 2a, 3a, 4a, …, (p-1)a gelijk zijn aan de getallen 1, 2, 3, 4, … p – 1
* De vermenigvuldiging van a \* 2a \* 3a \* (p – 1)a hetzelfde oplevert als 1\*2\* 3\*4\*, …\* p-1.
* ap-1 \* 1 \* 2 \* 3\* …. \* (p – 1) ☰ 1 \* 2 \* 3 \* … \* (p – 1) (mod p)

= ap-1 \* (p – 1) ☰ (p – 1) (mod p)

* Dit betekent dat p een deler is van ap-1 \* (p – 1) – (p – 1) = (ap-1 – 1)\* (p – 1) en omdat p geen deler is van (p – 1), geldt dat p een deler is van ap-1 – 1. Dit geeft weer de Kleine Stelling van Fermat namelijk, a­p-1 ☰ 1 (mod p).

*Voorbeeld 13:*

a= 3 en p = 13. Bewijs dat ap – 1 1 ☰ (mod p).

(13 – 1) \* 3 ☰ (13 – 1) (mod 13)

ap-1 \* (p – 1) ☰ (p – 1) (mod p)

Dit invullen:

313 – 1 (13 – 1) ☰ (13 – 1) (mod 13)

312 \* 12 ☰ 12 (mod 13)

312 ☰ 1 (mod 13)

= a­p-1 ☰ 1 (mod p)

### De Euler φ-functie

Voor de Euler φ-functie is het kennen van de term relatief priem van belang. Relatief priem betekent dat de grootste gemeenschappelijke deler tussen a en b, 1 is.

*Voorbeeld 14:*

a= 8 en b= 21

8 kan gedeeld worden door: 1, 2, 4 en 8

21 kan gedeeld worden door: 1, 3, 7 en 21

Het enige getal dat overeenkomt bij beide getallen is 1. Dit betekent dat de grootste gemeenschappelijke deler van 8 en 21, 1 is. Dus 8 en 21 zijn relatief priem aan elkaar.

De notatie van de Euler φ-functie wordt als volgt weergegeven φ(n) met n een natuurlijk positief geheel getal. Φ(n) is gelijk aan het aantal getallen dat relatief priem is met n. De Euler φ-functie werkt als volgt:

Stap 1: alle getallen kleiner dan n noteren, {1, 2, 3, …, n – 1}

Stap 2: vervolgens wordt gekeken of de grootste gemeenschappelijke deler tussen a (n) en b (een getal dat kleiner is dan n), 1 is.

Stap 3: het totale aantal getallen dat kleiner is dan n dat relatief priem is aan n, is dus de oplossing.

*Voorbeeld 15:*

φ(6)

Stap 1: alle getallen kleiner dan 6 opschrijven

dus 5, 4, 3, 2, 1

Stap 2: vervolgens wordt gekeken of de grootste gemeenschappelijke deler tussen 6 en b (een getal dat kleiner is dan 6), 1 is.

*6 kan gedeeld worden door 1, 2, 3 en 6*

5 kan gedeeld worden door 1 en 5; ggd = 1

4 kan gedeeld worden door 1, 2 en 4; ggd = 2

3 kan gedeeld worden door 1 en 3; ggd = 3

2 kan gedeeld worden door 1 en 2; ggd = 2

1 kan gedeeld worden door 1; ggd = 1

Stap 3: het totale aantal getallen dat kleiner is dan 6 dat relatief priem is aan 6, is dus de oplossing.

In dit geval zijn de 5 en de 1 relatief priem aan 6, dus 2 getallen

Oplossing:

φ(6)= 2

Wanneer p een priemgetal is dan geldt, φ(p)= (p – 1). p – 1 is het aantal getallen dat kleiner is dan p dat relatief priem is aan p

*Voorbeeld 16:*

φ(7), 7 is een priemgetal

*7 kan gedeeld worden door 1 en 7*

6 kan gedeeld worden door 1, 2, 3 en 6; ggd= 1

5 kan gedeeld worden door 1 en 5; ggd= 1

4 kan gedeeld worden door 1, 2 en 4; ggd= 1

3 kan gedeeld worden door 1 en 3; ggd= 1

2 kan gedeeld worden door 1 en 2; ggd= 1

1 kan gedeeld worden door 1; ggd= 1

Dus:

φ(7)= 6

Wanneer n uit 2 priemgetallen bestaat p en q geldt het volgende: φ(pq) = (p – 1) (q – 1). Want zoals hierboven geschreven is φ(p) = p – 1. En omdat n gelijk is aan pq, waarbij p en q elk een priemgetal is, is dat gelijk aan φ(pq) = φ(p)\* φ(q) en dat heeft als oplossing (p – 1) (q – 1).

De stelling van Euler zegt het volgende: laat n een positief geheel getal zijn en veronderstel dat a een geheel getal is dat relatief priem is met n, dan geldt dat a­­φ(n) ☰ 1 (mod n). Deze stelling komt overeen met de kleine stelling van Fermat a­p-1 ☰ 1 (mod p), want φ(p) = p – 1.

### Het RSA-algoritme

Om de werking van het RSA-systeem te begrijpen is uiteindelijk het RSA-algoritme nodig. Voor het RSA-algoritme zijn al de hiervoor behandelde termen nodig.

Stap 1: een getal kiezen voor p, q en k waarbij p en q priemgetallen zijn die het product vormen van n (n = pq) en k relatief priem aan φ(n). n wordt hier de enciphering modulus genoemd en k is het exponent modulus. Het paar (n, k) wordt door de boodschap verstuurder gepubliceerd.

De ontvanger wil naar de verstuurder het getal M sturen waarbij M< n. Wanneer M groter of gelijk is aan n, dan wordt het in stukjes geknipt door de ontvanger zo dat elk stukje kleiner is dan n. Daarna neemt hij het gepubliceerde paar (n, k) om r uit te rekenen met zijn computer, er geldt nu: Mk ☰ r (mod n)

*Voorbeeld 17:*

De verstuurder heeft p = 641, q = 701 en als enciphering exponent k = 3 gekozen. De boodschap is het woord NU. Toon dit aan.

***Stap 1:***

Modulo berekenen.

Modulo= n= pq

pq= 641 \* 701 = 449341

n= 449341

***Stap 2:***

Φ(pq) berekenen

Φ(pq) = (p – 1)(q – 1)

Φ(449341) = (641 – 1)(701 – 1)

Φ(449341) = 640\*700

Φ(449341) = 448000

***Stap 3:***

M bepalen.

A = 0, B = 1, C = 3, enzovoort, N= 13 en U= 20

M = 1320

***Stap 4:***

r berekenen

gebruik maken van Mk ☰ r (mod n)

13203 ☰ r (mod 449341)

r = 240762

Nu kan de ontvanger het getal r naar de verstuurder sturen via zijn e-mail bijvoorbeeld. De afluisteraar kan r wel onderscheppen en k en n kan hij opzoeken aangezien de verstuurder die in het openbaar gepubliceerd heeft. Maar M bepalen wordt voor hem wel lastig, als de priemgetallen hele grote getallen zijn, aangezien hij de priemgetallen niet kent.

Nadat de ontvanger zijn bericht heeft gestuurd naar de verstuurder, kan de verstuurder met de middelen die hij heeft, de boodschap ontcijferen. Met de priemfactoren en de enciphering exponent k, kan hij j bepalen en op die manier het bericht ontcijferen. Hierbij wordt gebruik gemaakt van het algoritme van Euclides; ax + by = ggd(a,b), waarbij a=k en b= φ(n) en ggd(k, φ(n))= 1

***Stap 5:***

j bepalen.

j = x, de vergelijking wordt:

kj + φ(n) y = ggd(k, φ(n)) = 1

kj ☰ 1 (mod(φ(n)) = 1 (mod (p – 1)(1 –q))

3j ☰ 1 (mod 448000)

j bepalen door het algoritme van Euclides achterstevoren uit te voeren. Dus,

3j + 448000y = 1

448000/3 = 149333 rest 1

448000 = 149333\*3 + 1

-149333\*3 = 1 – 448000

x = j = -149333

negatieve waarde is onhandig, daarom in links en rechts van -149333\*3 = 1 – 448000, 3 φ(n), optellen. Dus

-149333\*3 + 3\*448000 = 1 – 448000 + 3\*448000

896001 = 896001

3j = 896001

j = 298667

***Stap 6:***

t vinden.

Uit kj ☰ 1 (mod(φ(n)) volgt dat kj – 1 deelbaar is door φ(n), oftewel de volgende vergelijking ontstaat: kj = 1 + t φ(n), deze formule zal de verstuurder, die een bericht heeft ontvangen van de ontvanger, helpen om M te ontcijferen.

kj = 1 + t φ(n)

3\*298667 = 1 + t\* 448000

t= 2

***Stap 7:***

M vinden.

Vanwege de stelling van Fermat geldt het volgende: rj ☰ M \* (1)t ☰ M (mod n), nu kan de boodschap worden ontcijferd.

r = 240762

j = 298667

n= 448000

240762298667 ☰ M (mod 448000)

Omdat dit lastig is voor een rekenmachine om in te vullen, moet j vereenvoudigd worden.

|  |  |  |
| --- | --- | --- |
| ☰ 3621 (mod n) | ☰ 80752 (mod n) | ☰ 48912 (mod n) |
| ☰ 92260 (mod n) | ☰ 41037 (mod n) | ☰ 354642 (mod n) |
| ☰ 402264 (mod n) | ☰ 94117 (mod n) | ☰ 150556 (mod n) |
| ☰ 102391 (mod n) | ☰ 342010 (mod n) | ☰ 188344 (mod n) |
| ☰ 237091 (mod n) | ☰ 32522 (mod n) | ☰ 381111 (mod n) |
| ☰ 160140 (mod n) | ☰ 30048 (mod n) | ☰ 156235 (mod n) |

298667 = 218 + 215 + 211 + 210 + 29 + 27 + 25 + 23 + 2 + 1

Dus:

rj = \* \* \* \* \* \* \*\* \*

Dit invullen in rekenmachine geeft M = 1320.

# In hoeverre is het RSA-systeem veilig?

De werking van het RSA-systeem en het RSA-algoritme zijn ingewikkeld uit te voeren. Maar betekent dit dat ze ook ingewikkeld te kraken zijn? En wat betekent het kraken van het RSA-systeem voor de mensheid?

Hoewel het RSA-systeem gebruik maakt van grote priemfactoren die moeilijk te achterhalen zijn, hebben cryptografen en andere codekrakers in de loop der jaren toch wel een manier gevonden om die grote priemfactoren te vinden en hierdoor hebben ze ook de boodschap tussen de zender en de ontvanger weten te decoderen. Natuurlijk is het kraken van de boodschappen niet met pen en papier gebeurd, computers hebben een cruciale rol gespeeld bij het kraken van het RSA-systeem. Een aantal methodes die gebruikt worden om het RSA-systeem te kraken zijn: Trial division, brute force hacking en Number Field Sieve.

## Trial division

Trial division is het ontbinden van n in 2 priemgetallen. Dit gebeurt met behulp van grote supercomputers. Met processoren in die supercomputer, wordt er geprobeerd om die 2 priemgetallen te achterhalen. Met normale computers, kan het wel maanden duren totdat de priemgetallen gekraakt zijn, want er is bewezen dat de priemgetallen oneindig doorlopen en wanneer de gebruikte priemgetallen grote getallen zijn, is het zelfs voor de computer moeilijk om die getallen te berekenen. Deze methode is dus meer gebruikelijk voor n-getallen die niet uit te grote priemgetallen bestaan, dan zal het ontbinden van n niet heel lang duren.

## Brute force hacking

Brute force hacking is een veel gebruikte methode door hackers. Met behulp van deze methode probeert de hacker met zijn computer verschillende combinaties van mogelijke priemgetallen n te ontbinden totdat hij de juiste combinatie heeft gemaakt en de boodschap kan kraken. Deze methode duurt echter heel erg lang, maar het vinden van eenvoudige priemgetallen zal wel haalbaar zijn. Met het RSA-systeem is het ontbinden van n met de brute force al bijna onmogelijk, aangezien n bij het RSA-systeem uit hele grote priemgetallen bestaat. Deze methode is dus inefficiënt bij het kraken van het RSA-systeem.

## Number field sieve

Number Field Sieve betekent letterlijk getallenlichamenzeef, het is het snelste bekende algoritme voor de ontbinding van positieve natuurlijke getallen die uit meer dan 100 decimalen bestaan in priemfactoren. Deze methode is ontdekt in 1988 door de Britse wiskundige John Pollard. Hij heeft hiermee het zevende Fermat getal ontbonden. Een Fermat getal heeft de vorm Dit getal zou zoals Fermat vermoedde een priemgetal moeten zijn, maar dat is onjuist gebleken. In de loop der jaren werd deze methode verder uitgebreid door onder andere, Arjen en Hendrik Lenstra. Door deze wiskundigen is het NFS-algoritme de snelste methode geworden om cijfers te ontbinden.

In februari 1999 was er nieuw record behaald bij de ontbinding van n van het RSA-algoritme dat uit 140 decimalen bestond met behulp van het getallenlichamenzeef-algoritme. Eerder op 10 april in 1996 was ook een record behaald bij de ontbinding van n dat uit 130 cijfers bestond. Echter duurde de periode dat het eerste record werd gemaakt van het 130-cijferig-RSA twee keer langer dan de factorisatie van het 140-cijferig-RSA.

In feite is het de wiskundige weliswaar gelukt om 512-bit RSA te kraken en daarom is het gebruikt van de 512 bit nu niet meer veilig. Het kraken van het RSA-systeem betekent dat de persoonlijke informatie van de mensen niet meer veilig staat en dat mensen niet meer veilig met elkaar kunnen communiceren zonder dat er een afluisteraar is die ook geïnteresseerd is in dat gesprek. Daarom is er bedacht om het aantal bits van het RSA te verdubbelen en zo ook de priemgetallen te vergroten omdat n ook met 21024 wordt vergroot. Uiteindelijk is het in 2010 weer gelukt om dit systeem te kraken, hierdoor wordt er nu gebruik gemaakt van 22048 bits. Zo zorgt men ervoor dat het RSA altijd een stap voor loopt voor de RSA-hackers en kunnen mensen nog steeds veilig internetten. Maar dit geldt alleen voor software computers. Of dit ook zal gelden voor de kwantumcomputers, is nog de vraag.

# Wat is de invloed van kwantumcomputers op de cryptografie?

Encryptiemethoden zijn ontwikkeld om informatie alleen begrijpelijk te maken voor de ontvanger. Dit wordt gedaan door verschillende methoden zoals RSA of DAS, de twee meest verspreide encryptiemethoden op dit moment. Al deze encryptiemethoden zijn niet 100% veilig, er bestaat namelijk altijd een kans dat de sleutel gekraakt kan worden. De methoden zijn wel zo gemaakt dat dit niet eenvoudig is en dit niet binnen een redelijk tijdsbestek kan gebeuren.

## Huidige encryptiemethoden

Door de opkomst van de kwantumcomputer de afgelopen jaren, is de encryptie in groot gevaar. Kwantumcomputers kunnen enorm snel berekeningen uitvoeren, waar ‘normale’ computers miljoenen jaren over doen, kan een kwantumcomputer dit in een aantal minuten doen. Dit zorgt ervoor dat het mogelijk is dat de encryptiemethoden die nu dagelijks gebruikt worden, later niet meer veilig zijn en zelfs helemaal niet meer gebruikt kunnen worden, omdat de veiligheid van gegevens niet gegarandeerd kan worden.

In de cryptografie bestaan er twee soorten van encryptie, zo genoemde symmetrische- en asymmetrische encryptiemethoden. Bij beide methoden wordt er gebruik gemaakt van het coderen en decoderen van berichten, door middel van een sleutel. Het verschil tussen de twee soorten is dat er bij symmetrische encryptie gebruik gemaakt wordt van één sleutel, zowel voor het versleutelen en voor het decoderen. De sleutel moet dan wel op een veilige plaats zijn uitgewisseld, waar geen onderschepping mogelijk is. Dit is dus vaak bij een ontmoeting tussen de twee personen in een privé omgeving. Daarbij is deze encryptiemethode alleen eenvoudig, snel en betrouwbaar als er maar twee partijen in het spel zijn. Als er meer partijen in het spel zijn is het niet meer te achterhalen wie de verzender is of er zouden meer sleutels uitgewisseld moeten worden, alleen al bij kleine groepen zijn dit er al veel. Om het aantal sleutels te bepalen die nodig zijn kan de volgende formule opgesteld worden.

Hierbij in het aantal partijen

*Voorbeeld 18:*

Er zijn twee groepen, 1 en 2. Groep 1 bestaat uit 5 personen en groep 2 bestaat uit 10 personen. Om te bewijzen dat er veel sleutels uitgewisseld moeten worden bij kleine en grote groepen kan de formule gebruikt worden.

Groep 1:

= 5 = 10 sleutels

Groep 2:

= 10 = 45 sleutels

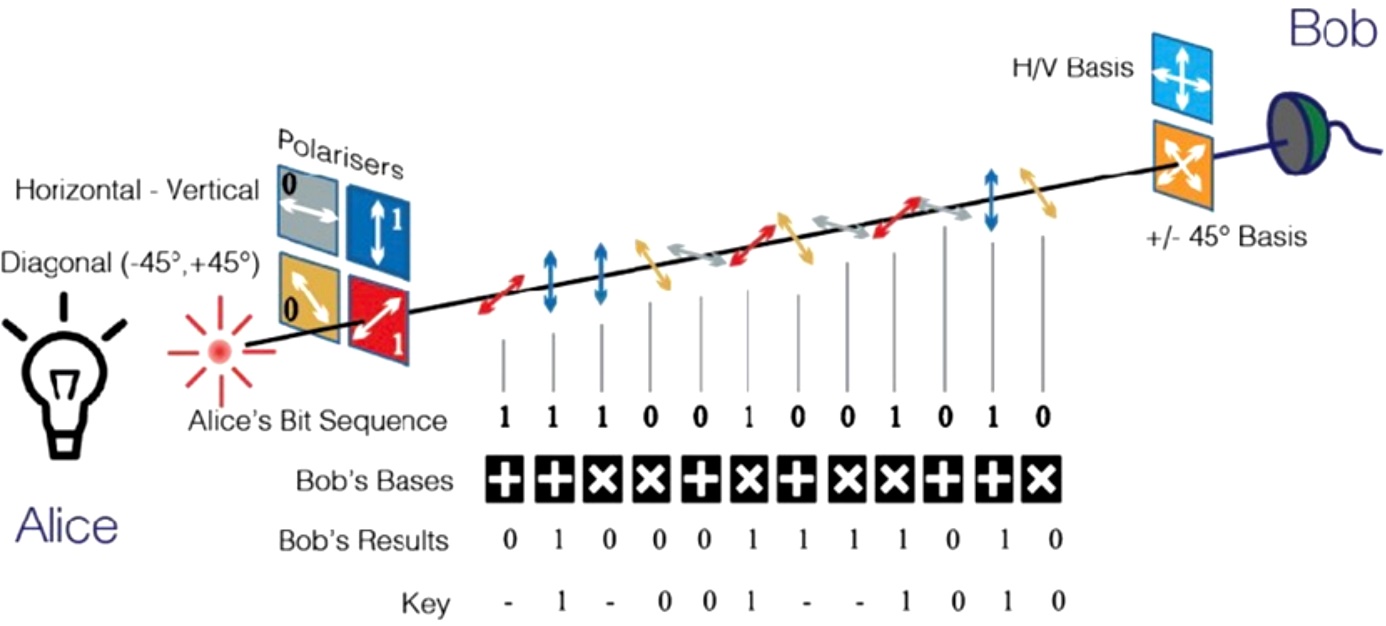
Symmetrische encryptiemethoden zijn dus niet handig voor groepen, hierbij wordt dan ook vaak overgestapt naar asymmetrische encryptiemethoden. Dit zijn encryptiemethoden met twee sleutels, een publieke sleutel en een privé sleutel. Hierbij hebben de zender en ontvanger beiden twee dezelfde sleutels, een die dus openbaar is en een die geheim is. De berichten die worden met de publieke sleutel beveiligd en kunnen alleen met de geheime sleutel worden ontcijferd. Het nadeel hiervan is, is dat er grote sleutellengtes gebruikt moeten worden en dat ook de geheime sleutel afgeleid kan worden van de publieke sleutel.

Een van de asymmetrische encryptiemethoden is de Diffie-Hellman encryptiemethode. Eerder, in de deelvraag ‘Hoe werkt het RSA-systeem’ onder het kopje ‘Diffie en Hellman’, is dit uitgelegd. Deze methode maakt gebruik van een relatief eenvoudige functie. Er bestaat op dit moment nog geen specifiek algoritme voor de klassieke computer om deze vergelijking op te lossen als de kraker het bericht onderschept heeft. Het is op moment dus alleen nog mogelijk om met logisch nadenken en gebruik van rekenmachines en andere algoritmes, de sleutel te kraken. Door de komst van de kwantumcomputer komt deze encryptiemethode in groot gevaar. Dat komt doordat er voor de kwantumcomputer een kwantumalgoritme bestaat die specifiek ontwikkeld is voor het ontbinden in factoren, dit algoritme wordt het algoritme van Shor genoemd. Dit is eerder uitgelegd in de deelvraag ‘Hoe werken kwantumcomputers?’ onder het kopje ‘Algoritmes’. Hierdoor duurt het kraken van de sleutels veel minder lang dan op een normale computer.

Advanced Encryption Standard (AES) is een encryptiemethode die gebruik maakt van 128 bits, boven de minimale eis die de NIST, een organisatie die de beveiliging van onlinegegevens waarborgt, stelt om een geaccepteerd encryptiesysteem te zijn. Dit is een encryptiemethode waarvan we nu al weten dat deze het kan opnemen tegen het kraken met een kwantumcomputer. Dit komt ten eerste omdat er maar één sleutel gebruikt wordt en de sleutel dus nergens uit afgeleid kan worden, wat wel bij asymmetrische encryptie mogelijk is. Daarbij komt ook, dat het algoritme een aantal keer opnieuw wordt uitgevoerd, met een sleutel die telkens iets anders is, en dat alleen de zender en ontvanger weten hoe vaak. Doordat dit dus een symmetrische encryptiemethode is, kunnen er niet meerdere berekeningen tegelijkertijd worden uitgevoerd, omdat de kwantumcomputer gebruik maakt van superpositie. Overigens is bij deze encryptiemethode het vorige antwoord nodig is om de volgende berekening te maken, terwijl de kraker nog helemaal niet zeker is dat de vorige berekening klopt.

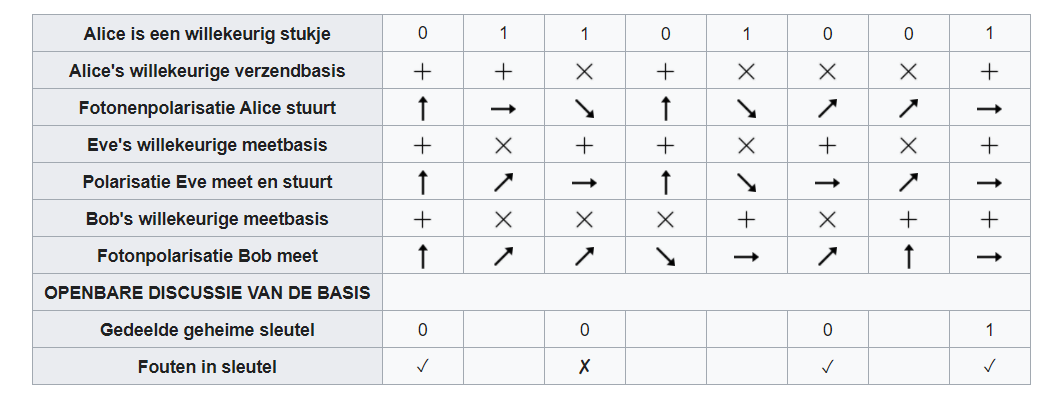
## Kwantumcryptografie

Een andere mogelijkheid is om juist de kwantumcomputer zelf te gebruiken om gegevens te beveiligen. Dit wordt ook wel kwantumcryptografie genoemd. Bij kwantumcryptografie wordt er gebruik gemaakt van de kwantumfysica om de berichten te versleutelen en te decoderen. Een methode die gebruikt wordt voor het genereren van een sleutel heet ‘Quantum Key Distribution’ (QKD). Hierbij worden de sleutels verstuurd door middel van fotonen, een van de elementaire deeltjes. In figuur 5 valt de werking van deze methode te zien. Alice en Bob doen dit via een speciaal kanaal gemaakt voor het overdragen van kwantuminformatie, een kwantumcommunicatiekanaal. Ook communiceren ze via een openbare klassieke zender, dit is bijvoorbeeld het internet, hier is dan wel van belang dat de identiteit geverifieerd is. Alice en Bob hebben beiden een basis, dit is een sjabloon, met twee richtingen (polarisatie) met een hoek van 90°, waar de fotonen doorheen kunnen. Alice kan met haar basis een van de 4 mogelijke polarisaties weergeven. Alice bepaalt willekeurig de polarisatie van het foton en stuurt dit naar Bob. Zij noteert welke basis zij heeft gebruikt (+ of x), de polarisatie (0 of 1) en de tijd van het versturen. Als Bob dit ontvangt kiest hij willekeurig een van de basissen en schrijft ook op welke basis hij heeft gebruikt, welke polarisatie hij heeft gemeten en op welk tijdstip. Als alle metingen zijn gedaan overleggen Alice en Bob welke basissen ze hebben gebruikt. Bij de basissen die overeenkomen klopt de uitkomst, hierdoor is de sleutel ontstaan.



Figuur 5 Quantum Key Distribution

Dit klinkt natuurlijk allemaal heel onveilig, want iemand kan zo de fotonen onderscheppen en deze aflezen. Dit klopt ook, maar er valt bijna altijd te achterhalen of er iemand heeft geïnterfereerd. Zoals eerder is uitgelegd, is dat in de kwantummechanica de polarisatie van het deeltje pas bepaald wordt, als er wordt geïnterfereerd. Degene die de sleutel wil kraken noemen we Eve. Op het moment dat zij interfereert kiest zij een willekeurige basis, want zij weet niet in welke basis Alice het foton heeft gestuurd. Als zij de goede basis kiest, krijgt zij het juiste resultaat en gaat de juiste polarisatie naar Bob toe. In dit geval valt de onderschepping moeilijk tot niet te achterhalen. Als Eve de verkeerde basis kiest, heeft zij dus 50% kans op het juiste resultaat. De polarisatie van dit foton stuurt Eve weer door naar Bob. Als Bob dit foton dan ontvangt en in dezelfde basis meet als die van Alice heeft hij, door de interferentie van Eve, 50% kans op de juiste polarisatie. Dit zorgt ervoor dat er fouten ontstaan in de sleutel. Pas op het moment dat Alice en Bob de sleutels met elkaar vergelijken (dan is de sleutel dus niet meer geheim!), wordt er duidelijk of Eve geïnterfereerd heeft.



Figuur 6 Interferentie van Eve

## Toekomst

Het is nog niet duidelijk wanneer de kwantumcomputer de huidige encryptiemethoden zal kraken, sommige ontwikkelaars, die kwantumcomputers bouwen, zeggen dat het binnenkort zal gebeuren. Terwijl andere onderzoekers zeggen dat dit helemaal niet mogelijk is. Het is nooit duidelijk wat er in de toekomst zal gebeuren, het is dus altijd maar een voorspelling. Wat wel vrijwel zeker is, is dat de encryptiemethoden steeds complexer worden en voortdurend verbeterd worden, alleen al om ervoor te zorgen dat gegevens beveiligd zijn tegen een mogelijke vijand. Ook de normale computer, blijft steeds beter worden. Dit zorgt er natuurlijk voor dat er als het ware een wedloop ontstaat tussen de ontwikkelaars van de encryptiemethoden en de mensen die de encryptie proberen te kraken. Als de kwantumcomputer steeds beter blijft worden en uiteindelijk beter is dan de normale computer, is de kans zeer groot dat de mensen, die de encryptiemethoden willen kraken, gedeeltelijk of helemaal, overstappen naar de kwantumcomputer in plaats van de normale computer. Veel krakers die onderscheppen bijvoorbeeld nu al informatie en slaan dit op, totdat het mogelijk is om deze informatie te decoderen door middel van een kwantumcomputer.

# Conclusie

In dit onderzoek is gezocht naar een antwoord op de vraag ‘Wat is de invloed van kwantumcomputers op de veiligheid van het RSA-systeem?’. Dit is gebeurd door middel van een analytisch bronnenonderzoek. Onze hypothese is dat kwantumcomputers de veiligheid van het RSA-systeem verslechteren.

Uit de resultaten van het onderzoek is gebleken dat kwantumcomputers op dit moment, niet sterk genoeg zijn om grote getallen te ontbinden in priemfactoren. Op dit moment kan een kwantumcomputer wel al kleinere getallen ontbinden in priemfactoren door middel van het algoritme van Shor. Er is wel een grote kans dat dit ooit mogelijk wordt in de toekomst, omdat de ontwikkeling van de kwantumcomputer in een rap tempo plaats kan gaan vinden.

Daarbij kan het RSA-algoritme ook op kwantumcomputers worden uitgevoerd, waardoor er grotere sleutels berekend worden. Een kwantumcomputer kan geen gebruik maken van de decodeermethode Number Field Sieve. Deze methode wordt wel gebruikt op een normale computer en kan een RSA-systeem, dat uit 512 bits bestaat, decoderen. Door de enorm snelle ontwikkeling van de normale computer wordt er nu gebruik gemaakt van een RSA-systeem met 2048 bits. Een kwantumcomputer heeft het vermogen om een RSA-systeem van het dubbel aantal bits te kraken. Op dit moment heeft de sterkste kwantumcomputer 16 kwantum bits, dus de sterkste kwantumcomputer kan nu een RSA-systeem van 32 bits kraken.

Een kwantumcomputer heeft natuurlijk de eigenschap dat dit meerdere berekeningen tegelijkertijd kan uitvoeren. Dit betekent dat de kwantumcomputer in een aantal opzichten veel sneller is dan de normale computer. Doordat het RSA-systeem gebruik maakt van een publieke sleutel kan een kraker makkelijk modulus *n* achterhalen. Als *n* dan bekend is, kan het algoritme van Shor toegepast worden, om de priemfactoren *p* en *q* te berekenen.

De kwantumcomputer kan daarentegen natuurlijk ook juist gebruikt worden om gegevens te beveiligen. Dit kan gedaan worden door gebruik te maken van kwantumcryptografie. Dit zal de gegevens coderen en decoderen door gebruik te maken van de kwantumfysica.

Hieruit kan geconcludeerd worden dat de veiligheid van het RSA-systeem, op dit moment, niet is gedaald met de opkomst van de kwantumcomputer. In de toekomst kan dit nog veranderen en kan er een wedloop ontstaan tussen de krakers en de ontwikkelaars. Onze hypothese is dus weerlegd.

# Discussie en reflectie

Het maken van dit onderzoek is niet helemaal volgens plan verlopen. Door de coronamaatregelen, werd de inleverdatum van het profielwerkstuk telkens verschoven. Dit zorgde ervoor dat de planning ook een beetje verschoof. De samenwerking tussen ons verliep goed. Ieder had zijn eigen deelvragen, op de afgesproken tijd, af en we gaven elkaar kritisch feedback over het gemaakte werk. Wanneer iemand vastliep bij het maken van zijn deelvraag, hielpen we elkaar. Uiteindelijk zijn we bij elkaar gekomen om de conclusie te schijven. Hierbij had ieder gebruik gemaakt van de kennis die hij in bezit had over zijn eigen deelvragen. Hierdoor konden we de deelvragen met elkaar verbinden om op het antwoord te komen van de hoofdvraag.

Het maken van een profielwerkstuk over cryptografie heeft veel nauwkeurigheid en diepgang nodig. In hoeverre wij zijn geslaagd om de deelvragen en de hoofdvraag te beantwoorden is moeilijk te zeggen. Doordat cryptografie zo een groot en diep onderwerp is, vonden we dat we toch niet altijd geslaagd zijn om alles nauwkeurig uit te leggen. Bij veel uitleg is er een grotere kans dat we van het hoofdonderwerp afhaken. Cryptografie bestaat namelijk uit veel formules en stellingen, die allemaal met elkaar te maken hebben en die samen, op verschillende manieren, voor de codering en de decodering van de gegevens zorgen.

De werking van een kwantumcomputer en het algoritme van Shor, vonden wij moeilijk om te begrijpen. Dit komt omdat hierover weinig informatie te vinden was op het internet en de gevonden informatie vooral in het Engels was. Wat wij ook erg lastig vonden, was de complete uitleg geven over de bestaande decodeermethodes, zoals het Number Field Sieve. Deze methode is erg ingewikkeld en een complete uitleg geven over de werking ervan is niet helemaal gelukt.

Al met al, vinden we dat we er toch wel goed uitgekomen zijn met het beantwoorden van de hoofdvraag en deelvragen. Alhoewel we hier in eerste instantie niet volledig in geslaagd waren.

# Aanbeveling voor vervolgonderzoek

Tijdens het maken van dit profielwerkstuk zijn we op een vraag gekomen die helaas geen grote invloed zal hebben op het beantwoorden van de hoofdvraag. Namelijk, hoe groot zal de invloed van de vervanging van de normale computer, door de kwantumcomputer, voor de mensheid zijn? Een kwantumcomputer heeft zoals eerder geformuleerd, een andere werkwijze dan een klassieke computer. Het zal nog lang duren totdat de mensheid eraan gewend is om een kwantumcomputer te gebruiken. De mensheid heeft deze hedendaags nog niet nodig, omdat niet iedereen de privacy van anderen wil beperken door gebruik te maken van kwantumcomputers. Maar hoe groot die invloed is op de mensheid, is interessant om te onderzoeken.

# Bronnenlijst

A. (2010, 8 maart). *Onverwacht snelle kraak 1024-bits RSA-sleutel*. Geraadpleegd op 1 januari 2021, van <https://www.agconnect.nl/artikel/onverwacht-snelle-kraak-1024-bits-rsa-sleutel>

Aarts, E. (z.d.). *Het RSA algoritme*. Geraadpleegd op 21 september 2020, van <http://crypto.aartsvandertogt.nl/pdf/rsa.pdf>

Aaten, A., & Kraaikamp, C. (2013). *Zebra-reeks 35 - Verborgen boodschappen* (1ste editie). Utrecht, Nederland: Epsilon Uitgaven.

arXiv, E. T. F. T. (2020, 2 april). *How a quantum computer could break 2048-bit RSA encryption in 8 hours*. Geraadpleegd op 18 november 2020, van <https://www.technologyreview.com/2019/05/30/65724/how-a-quantum-computer-could-break-2048-bit-rsa-encryption-in-8-hours/>

Bastiaansen, P. (2018, 29 maart). *Bekijk: Kwantumcomputers*. Geraadpleegd op 28 september 2020, van <https://www.nemokennislink.nl/publicaties/kwantumcomputers-1/>

Brandhof, A. V. D. (2019, 27 september). *Bekijk: Cryptografie*. Geraadpleegd op 28 september 2020, van <https://www.nemokennislink.nl/publicaties/cryptografie/>

Capgemini Nederland B.V. (2018, 23 mei). *Kwantumcomputers kraken versleuteling*. Geraadpleegd op 17 december 2020, van <https://www.trendsinveiligheid.nl/rapport/2016-de-mens-en-digitale-veiligheid/kwantumcomputers-kraken-versleuteling/>

Cavallar, S. (1999, 14 november). *Factorization of RSA-140 Using the Number Field Sieve*. Geraadpleegd op 21 december 2020, van <https://link.springer.com/chapter/10.1007/978-3-540-48000-6_16>

Dalen, B., & Puite, Q. (2009 september). Modulorekenen bij de IMO. *PYTHAGORAS*, *2009*(September). Geraadpleegd van https://www.wiskundeolympiade.nl

De Redactie. (2020, 30 augustus). *Begrijp kwantummechanica in vijf minuten*. Geraadpleegd op 28 september 2020, van <https://wibnet.nl/natuur/begrijp-kwantummechanica-in-vijf-minuten>

*DES aanvullend materiaal - DES supplementary material - qaz.wiki*. (z.d.). Geraadpleegd op 22 december 2020, van <https://nl.qaz.wiki/wiki/DES_supplementary_material#Initial_permutation_(IP)>

Ghose, S. (2018 november). *A beginner’s guide to quantum computing*. Geraadpleegd op 23 september 2020, van <https://www.ted.com/talks/shohini_ghose_a_beginner_s_guide_to_quantum_computing>

Hofstede, H. (2020 april). *Modulo rekenen*. Geraadpleegd op 29 september 2020, van <https://www.hhofstede.nl/modules/modulo.htm>

*How do we input data in quantum computers? - Quora*. (z.d.). Geraadpleegd op 14 december 2020, van <https://www.quora.com/How-do-we-input-data-in-quantum-computers>

*How does inputting data on quantum computers work (to be processed/analyzed)? - Quora*. (z.d.). Geraadpleegd op 14 december 2020, van <https://www.quora.com/How-does-inputting-data-on-quantum-computers-work-to-be-processed-analyzed>

*How is data transmitted in quantum computing? - Quora*. (z.d.). Geraadpleegd op 14 december 2020, van <https://www.quora.com/How-is-data-transmitted-in-quantum-computing>

Huang, Y. (2020, 16 september). *Benchmarking Quantum State Transfer on Quantum Devices using Spatio-Temporal Steering*. Geraadpleegd op 15 december 2020, van <https://arxiv.org/abs/2009.07425>

Koninklijke Nederlandse Akademie Van Wetenschappen. (z.d.). *Wanneer vervangt de kwantumcomputer de klassieke computer? — KNAW*. Geraadpleegd op 11 november 2020, van <https://www.knaw.nl/nl/thematisch/de-nederlandse-wetenschapsagenda/materialen-en-techniek/wanneer-vervangt-de-kwantumcomputer-de-klassieke>

Können, T. (2009, 27 november). Temmen van de fluxqubit: Onbreekbare codekraker. *De ingenieur*, *2009*(19). Geraadpleegd van <https://www.deingenieur.nl>

Kraaijvanger, C. (2018, 27 maart). *De quantummechanica: onbegrijpelijk, maar o zo fascinerend*. Geraadpleegd op 28 september 2020, van <https://www.scientias.nl/kwantummechanica-onbegrijpelijk-o-zo-fascinerend/>

*Letterfrequentie engels - Google zoeken*. (z.d.). Geraadpleegd op 22 december 2020, van <https://www.google.com/search?q=letterfrequentie+engels&sxsrf=ALeKk00KMD84ICA0e5gynkpYp_LqeIkimw:1608676412610&source=lnms&tbm=isch&sa=X&ved=2ahUKEwi29YbR0uLtAhXHO-wKHfxXD9EQ_AUoAXoECAcQAw&biw=1366&bih=657#imgrc=APrVYFdc3UJehM>

Martin, K. (2019, 22 januari). *Waiting for quantum computing: Why encryption has nothing to worry about*. Geraadpleegd op 18 november 2020, van <https://techbeacon.com/security/waiting-quantum-computing-why-encryption-has-nothing-worry-about>

Nashilov, E. (2020, 14 juli). *Kwantumcomputers en cryptografie voor dummies*. Geraadpleegd op 12 november 2020, van <https://www.kaspersky.nl/blog/quantum-computing-vs-data-encryption/25667/>

Next Quantum. (2018, 22 februari). *Geschiedenis*. Geraadpleegd op 11 november 2020, van <https://nextquantum.wordpress.com/kwantum-informatica/geschiedenis/>

pclooptvast. (z.d.). *Geschiedenis van de computer, ’Wanneer en hoe ontstaan?* Geraadpleegd op 10 november 2020, van <https://www.pclooptvast.nl/de-geschiedenis-van-de-computer-en-het-ontstaan/>

*Quantum Key Distribution (QKD)*. (2018, 14 november). Geraadpleegd op 17 december 2020, van <https://qt.eu/discover-quantum/underlying-principles/quantum-key-distribution-qkd/>

Querejeta, I. (2016). *A study of the General Number Field Sieve and a development of a CT2 plug-in using YAFU*. Geraadpleegd van <https://pure.tue.nl/ws/portalfiles/portal/46944529/855108-1.pdf>

Ravenstein, W. (2019, 10 juli). *Bekijk: Eenvoudige geheimschriften*. Geraadpleegd op 28 september 2020, van <https://www.nemokennislink.nl/publicaties/eenvoudige-geheimschriften/>

Redactie. (2018, 22 november). *Kwantumcomputer bedreiging voor digitale veiligheid*. Geraadpleegd op 12 november 2020, van <https://www.kpn.com/zakelijk/blog/kwantumcomputer-bedreiging-voor-digitale-veiligheid.htm>

S. (2020, 7 december). *What is DES?: Understanding DES Algorithm and Operation*. Geraadpleegd op 22 december 2020, van <https://www.simplilearn.com/what-is-des-article>

Schrauwers, A. (2007, 24 september). *De kwantumcomputer kan veel - op een enkel terrein - SYNC.nl*. Geraadpleegd op 7 oktober 2020, van <http://sync.nl/de-kwantumcomputer-kan-veel-op-een-enkel-terrein/>

Scribbr. (2020, 15 juni). *Scribbr APA Generator - Genereer bronnen in APA-stijl*. Geraadpleegd op 1 januari 2021, van <https://www.scribbr.nl/bronnengenerator/apa/>

Timmer, M. (2000, 17 december). *Cryptografie-uitleg aan de hand van RSA*. Geraadpleegd op 19 december 2020, van <https://tweakers.net/reviews/189/8/cryptografie-uitleg-aan-de-hand-van-rsa-rsa-kraken.html>

T.Q. Team. (2020, 5 oktober). *Shor’s Algorithm*. Geraadpleegd op 16 oktober 2020, van <https://qiskit.org/textbook/ch-algorithms/shor.html>

Tsehaie, N., & Feng, J. (2010). *Is RSA-cryptografie nu veilig genoeg en wat betekent dit voor de toekomst van digitale beveiliging?* Geraadpleegd van <https://www.marnixreunisten.nl/wp-content/uploads/2010/06/rsa-pws.pdf>

Universiteit van Nederland. (2015a, 13 januari). *Hoe kan een kwantumcomputer overal inbreken? (2/5)*. Geraadpleegd op 28 september 2020, van <https://www.youtube.com/watch?v=Qe_QBSyx1Bc>

Universiteit van Nederland. (2015b, 14 januari). *Hoe kun je ervoor zorgen dat de NSA je e-mail niet kan lezen? (3/5)*. Geraadpleegd op 28 september 2020, van <https://www.youtube.com/watch?v=_2l1pRrO8oM>

Veenstra, S. (2018). *De impact van het kwantumalgoritme van Shor op het RSA-algoritme zoals voorgeschreven door NIST* (Bachelor scriptie). Radboud Universiteit. Geraadpleegd van <https://www.math.ru.nl/~bosma/Students/SanneVeenstraBSc.pdf>

*Wat is een brute-force attack? - RealHosting.nl*. (2019, 17 mei). Geraadpleegd op 19 december 2020, van <https://realhosting.nl/helpdesk/wat-is-een-brute-force-attack/>

*Wat is kwantumcryptografie en hoe daarvan te profiteren? - dynabook*. (2017, 23 oktober). Geraadpleegd op 17 december 2020, van <https://nl.dynabook.com/generic/toshibytes-blogpost12-quantum-cryptography/>

Wikipedia contributors. (2020a, 18 november). *Quantum key distribution*. Geraadpleegd op 17 december 2020, van <https://en.wikipedia.org/wiki/Quantum_key_distribution#Quantum_key_exchange>

Wikipedia contributors. (2020b, 14 december). *Trial division*. Geraadpleegd op 19 december 2020, van <https://en.wikipedia.org/wiki/Trial_division#Speed>

Wikipedia contributors. (2021, 22 januari). Multiplicative order. Geraadpleegd op 23 januari 2021, van <https://en.wikipedia.org/wiki/Multiplicative_order>

Wikipedia-bijdragers. (2015, 9 maart). *Fermatgetal*. Geraadpleegd op 21 december 2020, van <https://nl.wikipedia.org/wiki/Fermatgetal>

Wikipedia-bijdragers. (2016, 14 november). *Peter Shor*. Geraadpleegd op 28 september 2020, van <https://nl.wikipedia.org/wiki/Peter_Shor>

Wikipedia-bijdragers. (2018, 25 juli). *Calciet*. Geraadpleegd op 28 september 2020, van <https://nl.wikipedia.org/wiki/Calciet>

Wikipedia-bijdragers. (2019a, 23 oktober). *Kwantumcomputer*. Geraadpleegd op 29 september 2020, van <https://nl.wikipedia.org/wiki/Kwantumcomputer>

Wikipedia-bijdragers. (2019b, 1 november). *Getallenlichamenzeef*. Geraadpleegd op 21 december 2020, van <https://nl.wikipedia.org/wiki/Getallenlichamenzeef>

Wikipedia-bijdragers. (2020a, 31 januari). *Konrad Zuse*. Geraadpleegd op 11 november 2020, van <https://nl.wikipedia.org/wiki/Konrad_Zuse>

Wikipedia-bijdragers. (2020b, 19 maart). *Brute force (methode)*. Geraadpleegd op 19 december 2020, van <https://nl.wikipedia.org/wiki/Brute_force_(methode)>

Wikipedia-bijdragers. (2020c, 18 juni). *Rekenmachine*. Geraadpleegd op 10 november 2020, van <https://nl.wikipedia.org/wiki/Rekenmachine>

Wikipedia-bijdragers. (2020d, 16 augustus). *Symmetrische cryptografie*. Geraadpleegd op 17 december 2020, van <https://nl.wikipedia.org/wiki/Symmetrische_cryptografie>

Wikipedia-bijdragers. (2020e, 24 september). *Geschiedenis van de computer*. Geraadpleegd op 8 november 2020, van <https://nl.wikipedia.org/wiki/Geschiedenis_van_de_computer>

Wikipedia-bijdragers. (2020f, 13 oktober). *Encryptie*. Geraadpleegd op 17 december 2020, van <https://nl.wikipedia.org/wiki/Encryptie>

Wikipedia-bijdragers. (2020g, 19 oktober). *Modulair rekenen*. Geraadpleegd op 2 november 2020, van <https://nl.wikipedia.org/wiki/Modulair_rekenen>

Wikipedia-bijdragers. (2020h, 21 oktober). *Computer*. Geraadpleegd op 11 november 2020, van <https://nl.wikipedia.org/wiki/Computer#Geschiedenis>

Wikipedia-bijdragers. (2020i, 6 november). *Algoritme*. Geraadpleegd op 7 oktober 2020, van <https://nl.wikipedia.org/wiki/Algoritme>

Wildeman, S. (2005). *Cryptografie: Het RSA Cryptosysteem*. Geraadpleegd van <https://pws.phikwadraat.nl/pws/pws%20cryptografie.pdf>

WiskundeAcademie. (2013, 8 april). *Exponentiele groei - wat is exponentiele groei? - WiskundeAcademie*. Geraadpleegd op 28 september 2020, van <https://www.youtube.com/watch?v=d6p_M_7Qwjc>

## Bijlagen

1. Veenstra, S. (2018). De impact van het kwantumalgoritme van Shor op het RSA-algoritme zoals voorgeschreven door NIST (Bachelor scriptie). Radboud Universiteit. Geraadpleegd van <https://www.math.ru.nl/~bosma/Students/SanneVeenstraBSc.pdf>

# Logboek

|  |  |  |  |
| --- | --- | --- | --- |
| Datum | Wat gedaan | Aantal uren | |
| 19-05-2020 | Oriënteren op het onderwerp en een onderwerp uitkiezen | 2 | |
| 30-06-2020 | Hoofdvraag en deelvragen bedacht, verder verdiepen in onderwerp | 6 |
| 01-07-2020 | Hoofdvraag specifieker gemaakt en deelvragen bedacht, plan van aanpak gemaakt, hypothese bedacht | 2 | |
| 14-09-2020 | Gesprek met PWS-begeleider over voortgang en aanpak | 1 |
| 14-09-2020 | Opzet gemaakt | 3 | |
| 15-09-2020 | Opzet verder gemaakt, taakverdeling gemaakt, onderzoeksplan gemaakt | 1 |
| 16-09-2020 | Overleg over taakverdeling en opzet | 1 |
| 21-09-2020 | Bronnen gezocht, inleiding geschreven en voorwoord afgemaakt | 4 |
| 22-09-2020 | Inleiding afgeschreven | 1 | |
| 23-09-2020 | Bronnen gezocht en deelvraag beantwoord | 1 | |
| 23-09-2020 | Bronnen gezocht | 1 | |
| 28-09-2020 | Bronnen gezocht, deelvraag beantwoord | 8 | |
| 28-09-2020 | Bronnen gezocht, deelvraag beantwoord | 7 | |
| 29-09-2020 | Bronnen gezocht, voetnoten gezet, corrigeren tekst | 2 | |
| 29-09-2020 | Werk van May nalezen en corrigeren | 2 | |
| 07-10-2020 | Bronnen gezocht, deelvraag beantwoord | 3 | |
| 07-10-2020 | Bronnen gezocht, deelvraag beantwoord | 2 | |
| 16-10-2020 | Bronnen gezocht | 2 | |
| 02-11-2020 | Bronnen gezocht, deelvraag beantwoord | 2 | |
| 04-11-2020 | Deelvraag 1 afgerond, bronnen gezocht | 2 | |
| 10-11-2020 | Deelvragen beantwoord, bronnen gezocht | 3 | |
| 10-11-2020 | Deelvraag afgerond, bronnen gezocht | 3 | |
| 11-11-2020 | Bronnen gezocht, deelvraag 2 beantwoord, spellingcontrole gedaan, werk van May nagelezen en gecorrigeerd, bronvermelding gemaakt | 7 | |
| 11-11-2020 | Deelvraag beantwoord en bronnen gezocht | 2 | |
| 12-11-2020 | Bronnen gezocht, deelvraag beantwoord | 2 | |
| 18-11-2020 | Bronnen gezocht, deelvraag beantwoord | 1 | |
| 18-11-2020 | Bronnen gezocht, deelvraag beantwoord | 4 | |
| 19-11-2020 | Bronnen gezocht, deelvraag beantwoord | 4 | |
| 21-11-2020 | Bronnen gezocht, deelvraag beantwoord | 3 | |
| 24-11-2020 | Bronnen gezocht, deelvraag beantwoord | 2 | |
| 26-11-2020 | Werk gecorrigeerd | 2 | |
| 29-11-2020 | Werk gecorrigeerd | 3 | |
| 14-12-2020 | Bronnen gezocht | 1 | |
| 14-12-2020 | Bronnen gezocht en feedback gegeven aan Tess | 1 | |
| 15-12-2020 | Bronnen gezocht, deelvragen beantwoord, werk gecorrigeerd | 3 | |
| 15-12-2020 | Bronnen gezocht en deelvragen beantwoord | 1 | |
| 17-12-2020 | Deelvragen beantwoord, werk gecorrigeerd, bronnen gezocht | 7 | |
| 18-12-2020 | Bronnen gezocht, deelvraag beantwoord | 3 | |
| 19-12-2020 | Bronnen gezocht, deelvraag 4 afgerond en deelvraag 5 beantwoord | 7 | |
| 20-12-2020 | Deelvragen afgerond, bronnenlijst bijgewerkt | 3 | |
| 21-12-2020 | Bronnen gezocht, deelvraag 5 afgerond, werk gecorrigeerd | 3 | |
| 21-12-2020 | Voorwoord gecorrigeerd | 1 | |
| 22-12-2020 | Feedback meneer Rauwerda verwerkt, bronnenlijst verwerkt | 3 | |
| 28-12-2020 | Deel van de samenvatting geschreven | 2 | |
| 29-12-2020 | Deel van de samenvatting geschreven | 2 | |
| 01-01-2021 | Overleg gehad, conclusie geschreven, discussie geschreven, bronnenlijst gemaakt, werk gecorrigeerd | 4 | |
| 16-01-2021 | Werk gecorrigeerd | 1 | |
| 23-01-2021 | Werk gecorrigeerd | 3 | |
| 23-01-2021 | Werk gecorrigeerd | 4 | |
| 24-01-2021 | Werk gecorrigeerd | 3 | |
| 25-01-2021 | Werk gecorrigeerd en PWS afgerond! | 4 | |

# Bijlagen

Afbeelding met tekst

Automatisch gegenereerde beschrijving1.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

1. Veenstra, S. (2018). De impact van het kwantumalgoritme van Shor op het RSA-algoritme zoals voorgeschreven door NIST (Bachelor scriptie). Radboud Universiteit. Geraadpleegd van <https://www.math.ru.nl/~bosma/Students/SanneVeenstraBSc.pdf> [↑](#footnote-ref-2)