Figuur 1: RSA-decodering

Een geheime boodschap

Robin Aerts, Elias Beyen, Ward Vancoillie
Vak: Wiskunde
6C Wetenschappen-Wiskunde &
6A Grieks-Wiskunde

Sint-Andreasinstituut Oostende
2022-2023

Voorwoord

Voor het vak wiskunde moesten we een wiskundige toepassing gaan onderzoeken. Deze paper is geschreven zodat iedereen met wat basiskennis wiskunde hem kan volgen. Hij is dan ook vooral bedoeld voor geïnteresseerden in het vak wiskunde die graag eens willen zien hoe wiskunde wordt toegepast in onze moderne samenleving.

Het onderwerp dat ons direct aansprak was het onderwerp met de naam ‘Geheime code’. We hebben dan ook voor dit onderwerp gekozen. Tijdens het onderzoek probeerden wij een code, die aan de hand van RSA-codering was opgesteld, te ontcijferen. Het leek ons wel interessant om eens te kijken hoe deze vorm van codering werkt omdat het al op erg veel plaatsen in onze samenleving wordt gebruikt en het dus ook erg relevant is.

Tijdens dit onderzoek hebben we erg veel bijgeleerd. Op het eerste gezicht leek de wiskunde soms wat moeilijk, maar we wilden weten wat de code ons ging vertellen. Hierdoor bleven we doorzetten en uiteindelijk viel de wiskunde goed mee. De boodschap zal op het einde van deze paper ontcijferd worden. Wij zouden onze leerkracht wiskunde, mevrouw Duflou, graag bedanken voor haar begeleiding en hulp tijdens dit onderzoek.

Inleiding

We vroegen ons af hoe we een code, enkel bestaande uit cijfers, konden kraken. De code die ontcijferd zal worden, staat hieronder genoteerd:

59862661107273246467688391168087069044282709232523357652
15384960551644676685158733048529735129102653572420840534
71308478924940706858622927889905277610721589366702651537
13033720489310879088188625 41

We kregen slechts vier gegevens mee: drie getallen en een aangepast alfabet waarbij elke letter stond voor een ander getal. Hoe wij die code gekraakt hebben? We hebben het in dit onderzoek neergeschreven door middel van een literatuurstudie. Om dit te kunnen doen, moesten we ons wat verdiepen in verschillende wiskundige stellingen. Zo konden we een manier vinden om de code, die door middel van RSA gecodeerd was, te ontcijferen.

RSA is een wiskundige manier om gegevens te coderen. Om de toch wel ingewikkelde wiskunde hierachter te verstaan en zelf te kunnen gebruiken, moesten we eerst ons hoofd buigen over enkele andere wiskundige stellingen.

We begonnen met het modulorekenen, ook wel klokrekenen genoemd. Daarna probeerden we de kleine stelling van Fermat te begrijpen en vervolgens hebben we het in dit onderzoek even over de Eulerfunctie en de bijhorende stelling van Euler.

Na dit alles grondig te verwerken konden we uiteindelijk de manier waarop RSA toegepast wordt begrijpen en begonnen we uit te zoeken hoe we de grote code konden ontcijferen. Vooraleer we deze code probeerden te ontcijferen, stelden we zelf een kleinere code op en ontcijferden die, zodat we er zeker van waren dat we begrepen hoe het principe werkt. Tot slot ontcijferden we de grote code.

Inhoudstafel

Corpus

1 Modulorekenen

Modulorekenen of klokrekenen is handig om uren te berekenen. Het werkt als volgt: Als je in de "normale" wiskunde 20 bij 5 optelt bekom je 25, want 20 + 5 = 25. Bij modulorekenen werkt het iets anders. Veronderstel nu dat de 20 in het vorig voorbeeld 20u 's avonds wordt, als je 5u laat verlopen kom je op 1u 's nachts uit, niet op 25u omdat de uren van de dag elke dag opnieuw starten bij 0u. Het kleine, maar belangrijke verschil met een klok, is dat modulorekenen enkel en alleen met gehele getallen werkt.[1]

Een '=' noteren we in modulorekenen als een '$\equiv$'. We zullen het ook niet altijd even uitgebreid uitschrijven zoals in het voorbeeld hierboven, maar gebruiken een verkorte notatie. Met name deze:

$$z \equiv x \ (mod\ y)$$

Hierbij is het getal *y* de modulus, en *x* en *z* zijn congruent modulo *y*. Dat wil zeggen dat *x* en *z*, wanneer ze gedeeld worden door *y*, dezelfde rest hebben. Neem nu bijvoorbeeld $4 \equiv 11$ (mod 7). Als je hierbij 11 deelt door 7 bekom je een rest van 4, en als je nu 4 deelt door 7 bekom je ook een rest van 4.[2,3]

Soms worden de haakjes weggelaten. We gebruiken als conventie in deze paper het gelijkheidsteken '=' in plaats van '$\equiv$'.

[1] Hofstede, H. (Z.d.). *Modulo-rekenen*.
[15.03.2023, H.Hofstede: https://www.hhofstede.nl/modules/modulo.htm].
[2] Hofstede, H. (Z.d.). *Modulo-rekenen*.
[15.03.2023, H.Hofstede: https://www.hhofstede.nl/modules/modulo.htm].
[3] Z.a. (Z.d.). *Modulo rekenen*. [17.03.2023, Math4All: https://www.math4all.nl/venster/bekijk/modulo-rekenen/13/1].

2 Kleine stelling van Fermat

Pierre de Fermat leefde van 1607 tot 1665. Hij heeft enkele bijdragen geleverd voor de wetenschap, in dit geval de getaltheorie.[4]

Bij de kleine stelling van Fermat is de essentie deelbaarheid. Wanneer je twee natuurlijke getallen deelt kan het zijn dat het uitkomt, de rest is nul met andere woorden. Bijvoorbeeld als je het getal 20 deelt door 5 dan kom je 4 uit, hier is de rest dus nul. Het kan ook zijn dat het niet uitkomt en dan heb je een rest die niet gelijk is aan nul.[5]

De kleine stelling van Fermat luidt: als p een priemgetal is en a een natuurlijk getal die geen factor p bevat, dan is:

$$a^{p-1} = 1 \ (mod\ p)$$

Deze stelling maakt het mogelijk om snel en efficiënt grote machten modulo van een priemgetal te bepalen.[6] Stel dat men bijvoorbeeld 5^{301} (mod 11) wil berekenen, dan kan dit aan de hand van de kleine stelling van Fermat. In dit geval is a het getal 5 en p het priemgetal 11. Volgens Fermat is dan:

$$5^{10} = 1 \text{ (mod 11)}$$

We weten dat $5^{300} = (5^{10})^{30}$, dus 5^{300} (mod 11) = 1^{30} (mod 11) = 1 (mod 11). Men wil natuurlijk weten hoeveel 5^{301} (mod 11) is en dit kan met simpele wiskunde berekend worden:

$$5^{301} \text{ (mod 11)}$$
$$= 5^{300} . 5^{1} \text{ (mod 11)}$$
$$= 5 \text{ (mod 11)}$$
$$= 5$$

[4] Hofstede, H. (Z.d.). *De kleine stelling van Fermat*. [16.03.2023, H.Hofstede: https://www.hhofstede.nl/modules/fermatklein.htm].

[5] Vanhoorn, M. (Z.d.). *De kleine stelling van Fermat*. [15.03.2023, Rijksuniversiteit Groningen: https://www.math.rug.nl/~broer/pdf/ksFermat.pdf].

[6] Vanhoorn, M. (Z.d.). *De kleine stelling van Fermat*. [15.03.2023, Rijksuniversiteit Groningen: https://www.math.rug.nl/~broer/pdf/ksFermat.pdf].

3 Stelling van Euler

Leonhard Euler was een Zwiterse wiskundige die grote bijdragen heeft geleverd aan de wiskunde.

Een belangrijke stelling die vaak binnen de cryptografie wordt gebruikt, is de stelling van Euler. Deze stelling vergemakkelijkt, net zoals bij de kleine stelling van Fermat[7], het rekenen met grote machten. De stelling van Euler is dan ook een veralgemening van de kleine stelling van Fermat. Deze stelling is dus niet langer beperkt tot enkel priemgetallen. De stelling luidt als volgt: Voor elk geheel getal *a* en natuurlijk getal *n*, waarvoor geldt dat *a* en *n* relatief priem zijn, wat wil zeggen dat de grootst gemene deler gelijk is aan 1, geldt dat:[8]

$$a^{\varphi(n)} = 1 \ (mod\ n)$$

Hierbij is φ de functie van Euler, deze functie wordt ook wel de phi-functie of totiëntfunctie genoemd. Dit zijn het aantal natuurlijke getallen kleiner of gelijk aan *n* die onderling deelbaar zijn met *n*.[9] Zo is bijvoorbeeld $\varphi(8) = 4$, want er zijn 4 getallen, die kleiner zijn dan 8 en die als grootst gemene deler 1 hebben, met name 1, 3, 5 en 7.

[7] Zie supra
[8] De Naeghel, K. (21.05.2007). *RSA codering*. [16.03.2023, Koen De Naeghel: https://koendenaeghel.webs.com/RSA.pdf].
[9] Z.a. (Z.d.). *Security - deel 3*. [16.03.2023, Edictum: https://www.edictum.nl/lesson/security/3#].

4 De stelling van Bézout

De Bézoutcoëfficiënten zijn zeer belangrijk bij het ontcijferen van de code. Aan de hand van deze stelling kunnen we de waarde *d* berekenen die we later nodig zullen hebben. Wanneer je twee gehele getallen *a* en *b* hebt, dan zegt Bézout dat er twee gehele getallen *x* en *y* bestaan zodat:[10]

$$ax + by = ggd(a, b)$$

De waarden van *x* en *y* noemen we de Bézoutcoëfficiënten. Bij RSA-decodering zal de grootste gemene deler van *a* en *b* gelijk zijn aan 1. Hierbij geldt de volgende formule:[11]

$$ax = 1(mod\ b)$$

Wanneer *x* negatief is moet je bij het decoderen er *b* bij optellen totdat *x* positief is. Waarom we deze waarden nu nodig hebben, wordt later duidelijker.[12]

Stel nu *a*=3 en *b*=7, dan is de grootste gemene deler gelijk aan 1. Volgens Bézout bestaan *x* en *y* zodat:

$$3x + 7y = 1$$

Hier kan *x* gelijk zijn aan -2 en *y* aan 1. Hier zijn -2 en 1 de Bézoutcoëfficiënten. Je ziet dat dit niet de enige getallen zijn die kunnen voorkomen als *x* en *y*. De getallen -9 en 4 zijn even goede coëfficiënten. De grootste gemene deler van *a* en *b* is 1 dus hier geldt ook:

$$3.(-2) = 1(mod\ 7)$$

[10]Z.a. (Z.d.). *Bézoutcoëfficiënten*.
[18.03.2023, comnuan.com: https://comnuan.com/cmnn02/cmnn02007/cmnn02007.php].
[11] Z.a. (Z.d.). *Bézoutcoëfficiënten*.
[18.03.2023, comnuan.com: https://comnuan.com/cmnn02/cmnn02007/cmnn02007.php].
[12] Z.a. (Z.d.). *Bézoutcoëfficiënten*.
[18.03.2023, comnuan.com: https://comnuan.com/cmnn02/cmnn02007/cmnn02007.php].

5 RSA-codering

5.1 RSA uitgelegd

RSA is een encryptiealgoritme gemaakt in 1977 door Ron **R**ivest, Adi **S**hamir en Len **A**dleman. Dankzij dit algoritme kunnen bepaalde gegevens, aan de hand van twee grote priemgetallen, versleuteld worden. Het is een asymmetrisch algoritme, wat betekent dat het werkt aan de hand van twee sleutels: Een publieke sleutel en een geheime sleutel.[13] De verzender gebruikt de publieke sleutel om het bericht te versleutelen. Hierna verstuurt hij de nieuwe, versleutelde boodschap naar de ontvanger en deze kan de boodschap ontcijferen mits hij de geheime sleutel heeft.

Stel bijvoorbeeld dat ik een geheime pincode wil versturen naar mijn broer. Mijn pincode is 1234. Als ik deze zo over het internet zou versturen, bestaat de kans dat iemand de code onderschept. Om dit te voorkomen kunnen we met het RSA-algoritme de pincode versleutelen. Simpel uitgelegd voer ik 1234 samen met een publieke sleutel in het algoritme in. Hierna bekomen we een nieuwe code: bijvoorbeeld 3850. Deze code kan ik nu veilig versturen naar mijn broer en als hij onderschept zou worden, zou die persoon er niets mee zijn doordat hij de geheime sleutel niet bezit. Mijn broer kan dan aan de hand van de geheime sleutel die hij heeft, mijn code ontcijferen en dan bekomt hij opnieuw 1234.

5.2 Waarom we RSA nodig hebben

Zoals eerder vermeld is RSA een asymmetrisch algoritme. Het voordeel hierbij ten opzichte van een symmetrisch versleutelingsalgoritme is dat het twee sleutels nodig heeft om te werken. Doordat symmetrische algoritmen maar met één sleutel werken, wordt dezelfde sleutel gebruikt om zowel het bericht te versleutelen als te ontcijferen. Dit maakt RSA veel veiliger, maar ook wel veel trager. Aan de basis van RSA ligt namelijk het vermenigvuldigen van grote priemgetallen, wat erg veel rekenkracht vraagt wanneer deze priemgetallen uit honderden cijfers bestaan.[14]

Een code is er echter om gekraakt te worden. Er bestaan dan ook meerdere mogelijkheden om RSA te kraken. Dit duurt echter enorm lang, en hoe groter de priemgetallen zijn die werden gekozen, hoe langer dit duurt. Momenteel is RSA nog een redelijk veilige manier om boodschappen te versleutelen, maar naarmate computers steeds sterker worden in de toekomst, zal de tijd die nodig is om een RSA-boodschap te kraken ook steeds korter worden…

5.3 Toepassingen van RSA in het dagelijks leven

RSA wordt tot op de dag van vandaag nog steeds op enorm veel plaatsen gebruikt. Zo wordt het erg veel gebruikt op het internet en in je browser, maar ook bijvoorbeeld in VPN's en nog in vele andere communicatietoepassingen zoals bijvoorbeeld in SSH: Een protocol dat erg vaak gebruikt wordt

[13] Timmer, M. (17.12.2000). *Cryptografie-uitleg aan de hand van RSA*. [15.03.2023, Tweakers: https://tweakers.net/reviews/189/3/cryptografie-uitleg-aan-de-hand-van-rsa-algoritmes.html].
[14] Timmer, M. (17.12.2000). *Cryptografie-uitleg aan de hand van RSA*. [15.03.2023, Tweakers: https://tweakers.net/reviews/189/3/cryptografie-uitleg-aan-de-hand-van-rsa-algoritmes.html].

om veilig met andere computers verbinding te maken. Niet enkel in digitale toepassingen wordt RSA gebruikt, maar ook in chipkaarten, zoals bankkaarten of identiteitskaarten, wordt RSA gebruikt.[15]

Vaak wordt echter RSA gebruikt in combinatie met nog andere versleutelingssystemen. Zo wordt het steeds moeilijker om de boodschap te ontcijferen en verkleint de kans dat je gegevens in de foute handen vallen.

5.4 Zelf een code coderen en decoderen

Voordat we zullen uitleggen hoe je de grote code kunt decoderen, gaan we eerst bekijken hoe je zelf een code kunt coderen en dan terug decoderen. We zullen dit uitleggen aan de hand van een eenvoudig voorbeeld. Stel nu dat je het woord 'ei' wilt gaan coderen, dan is de eerste stap de letters omzetten naar een getal. Dit doen we aan de hand van een aangepast alfabet waarbij iedere letter een getal voorstelt.[16] Het woord 'ei' wordt dan het getal 1418.

A. Code coderen

Stap 1: keuze 2 priemgetallen

Ten eerste kies je zelf twee grote priemgetallen, hier kiezen we $p = 71$ en $q = 83$. Daarna bereken je het product n van de twee getallen, in dit geval wordt dit dan 5893.[17]

Stap 2: keuze getal *e*

Ten tweede kies je een natuurlijk getal e, dit mag niet zomaar een getal zijn. Het getal e moet voldoen aan enkele voorwaarden. Het getal e moet groter zijn dan 1, maar moet kleiner zijn dan het product van de twee gekozen priemgetallen. De grootste gemene deler van e en $\varphi(n)$ moet gelijk zijn aan 1. Hierbij is $\varphi(n)$ gelijk aan $(p\text{-}1).(q\text{-}1)$ doordat n gelijk is aan het product van twee priemgetallen. We komen 5740 uit voor $\varphi(n)$. We kiezen voor e het getal 1249. Als je de voorwaarden nog eens verifieert dan zie je dat 1249 kleiner is dan 5740. De grootste gemene deler van 5740 en 1249 is 1 dus dit is een goede keuze voor e. De twee priemgetallen en het getal e zijn de gegevens die men geeft bij een geheime code.[18]

Stap 3: Boodschap coderen

Nu kunnen we het getal 1418 omzetten naar de geheime code. Dit doe je aan de hand van modulorekenen. Om het getal om te zetten naar een geheime code moet je gebruik maken van volgende formule:

---

[15] Van Miltenburg, O. (19.03.2021). *Waar wordt RSA gebruikt en wie is Schnorr?*. [15.03.2023, Tweakers: https://tweakers.net/reviews/8834/2/de-vernietiging-van-rsa-encryptie-met-priemgetallen-nog-niet-ten-eind-waar-wordt-rsa-gebruikt-en-wie-is-schnorr.html].

[16] Zie infra figuur 2

[17] De Naeghel, K. (21.05.2007). *RSA codering*. [16.03.2023, Koen De Naeghel: https://koendenaeghel.webs.com/RSA.pdf].

[18] De Naeghel, K. (21.05.2007). *RSA codering*. [16.03.2023, Koen De Naeghel: https://koendenaeghel.webs.com/RSA.pdf].

$$c = b^e (mod\ n)$$

Hierbij is *c* de versleutelde code en *b* is de originele boodschap, namelijk het getal 1418. Als we deze formule toepassen:[19]

$$c = 1418^{1249}\ (mod\ 5893) = 4628$$

B. De gemaakte code decoderen

Stap 1: *d* Berekenen

Voordat we de code decoderen moeten we het getal *d* berekenen, de waarde *d* is essentieel bij het ontcijferen van zo'n code. Deze heeft ook enkele voorwaarden: Het moet tussen 1 en $\varphi(n)$ liggen en het product van *d* en *e* moet gelijk zijn aan 1 (mod $\varphi(n)$). Aan de hand van de stelling van Bézout komen we het getal 1489 uit voor *d*.

Stap 2: Het ontcijferen

Wanneer je een code gaat decoderen, krijg je de twee priemgetallen *p* en *q* en het getal *e* gegeven. Het getal *d* zal je dus zelf moeten berekenen. De code is 4628. De formule om deze code te ontcijferen kunnen we afleiden uit de versleutelingsformule:[20]

$c^d\ mod\ n = (b^e mod\ n)^d$	($c = b^e \pmod n$)
$= b^{e.d}\ mod\ n$	($(a^p)^q = a^{p.q}$)
$= b^{1+k\varphi(n)}\ mod\ n$	($d.e = 1 \bmod \varphi(n) \Rightarrow d.e - 1 = k.\varphi(n)$)
$= (b.b^{k\varphi(n)})\ mod\ n$	($a^{p+q} = a^p.a^q$)
$= b\ mod\ n\ .\ (b^{\varphi(n)} mod\ n)^k$	(Eigenschap modulorekenen)
$= b\ mod\ n$	(Stelling van Euler)
$= b$	($b < n$)

Toegepast is dit:

$$b = 4628^{1489}\ (mod\ 5893) = 1418$$

Hier zie je dat we opnieuw het getal 1418 uitkomen. Om dit te ontcijferen hoef je alleen nog maar te kijken naar het aangepaste alfabet en kom je opnieuw het woord 'ei' uit.

[19] De Naeghel, K. (21.05.2007). *RSA codering*. [16.03.2023, Koen De Naeghel: https://koendenaeghel.webs.com/RSA.pdf].
[20] De Naeghel, K. (21.05.2007). *RSA codering*. [16.03.2023, Koen De Naeghel: https://koendenaeghel.webs.com/RSA.pdf].

a	10	f	15	k	20	p	25	u	30	z	35
b	11	g	16	l	21	q	26	v	31	spatie	36
c	12	h	17	m	22	r	27	w	32		
d	13	i	18	n	23	s	28	x	33		
e	14	j	19	o	24	t	29	y	34		

Figuur 2: Het aangepaste alfabet

5.5 De geheime boodschap ontcijferen

Nu is het tijd om de gekregen geheime boodschap te ontcijferen. De versleutelde boodschap stellen we gelijk aan *c* en gaat als volgt:

c = 59862661107273246467688391168087069044282709232523357652
153849605516446766851587330485297351291026535724208405 34
713084789249407068586229278899052776107215893667026515 37
130337204893108790881886254 1

Daarnaast kregen we ook twee grote priemgetallen, die we hier definiëren als *p* en *q*:

p = 152415787806736785461057783115378780769699778427732407842211 41
018789041253356941487924841722494260 7

q = 152415787806736785461057783115378780769699778430818827570297 82
991597683226165583460733483695308831 7

Als laatste kregen we ook een vercijferingsexponent *e*:

e = 987654321

Ook stellen we *n* gelijk aan het product van *p* en *q*. Dit product kunnen we niet zomaar met een gewoon zakrekenmachine berekenen, want deze getallen zijn veel te groot. Daarom gebruiken we een online rekenmachine die deze grote getallen aankan.[21] Met al deze gegevens kunnen we nu de stappen doorlopen die eerder geformuleerd zijn.

Stap 1: $\varphi(n)$ berekenen

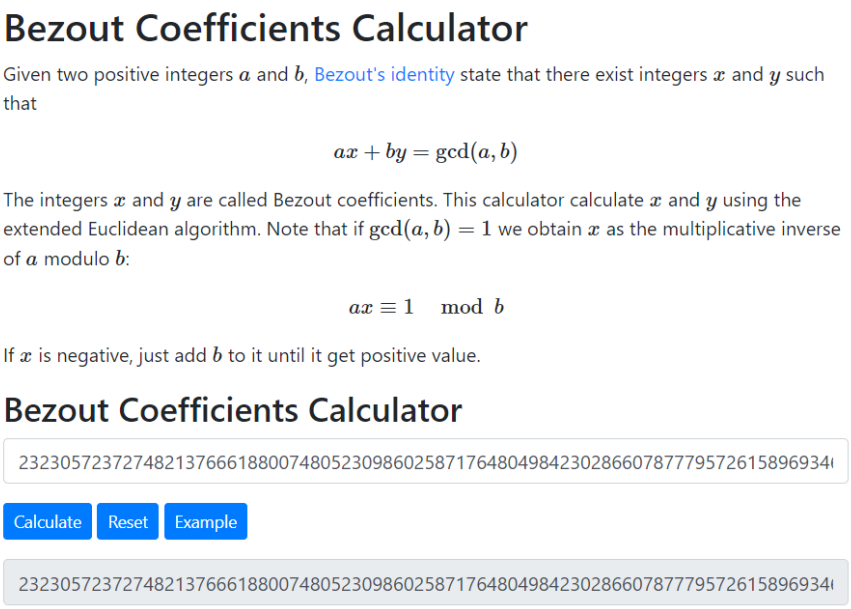
Aangezien *n* het product is van de twee priemgetallen *p* en *q*, geldt dat $\varphi(n) = (p-1)(q-1)$. Als we dit invullen in het online rekenmachine bekomen we het volgende getal:

$\varphi(n) =$
232305723727482137666188007480523098602587176480498423028660787779572615896
934611084447477046200896890258002327398413404750826418614537665292640123265
403832507294823457795500017140430085575491914 96

Stap 2: *d* berekenen

Aan de hand van de stelling van Bézout kunnen we *d* berekenen. Hiervoor gebruiken we een online rekenmachine die aan de hand van de twee getallen $\varphi(n)$ en *e*, de Bézoutcoëfficiënten kan berekenen.[22]

[21] https://web2.0calc.com/
[22] https://comnuan.com/cmnn02/cmnn02007/



Figuur 3: Bézoutcoëfficiënten rekenmachine

Het resultaat dat de rekenmachine geeft is: (Hierbij zijn de getallen *e* en $\varphi(n)$ vervangen door de corresponderende variabelen):

$\varphi(n)$. (83344825) + e . (-
196034983889523695552326226298795168856355084695784538364812189606677357377
979144155198154558644721062903114751083172454389654783526283829924373558848
972129132717056016361627751295649380417426411 9) = 1

In deze vergelijking zijn de getallen die vermenigvuldigd worden met *e* en $\varphi(n)$ de bézoutcoëfficiënten. Doordat de grootste gemene deler van *e* en $\varphi(n)$ gelijk is aan 1, geldt dat:[23]

$$e.d = 1\ mod(\varphi(n))$$

Hieruit kunnen we afleiden dat *d* het getal is waarmee *e* vermenigvuldigd wordt. Aangezien dit getal negatief is, moeten we hier $\varphi(n)$ bij optellen om een positief getal voor *d* te bekomen.

Na deze berekening bekomen we het volgende getal voor *d*:

d =
212702225338529768110955384850643581716951668010919969192179568818904880159
136696668927661590336424783967690852290096159311860940261909282300202767380
506619594023117856159337242010865147533749273 77

[23] Zie supra

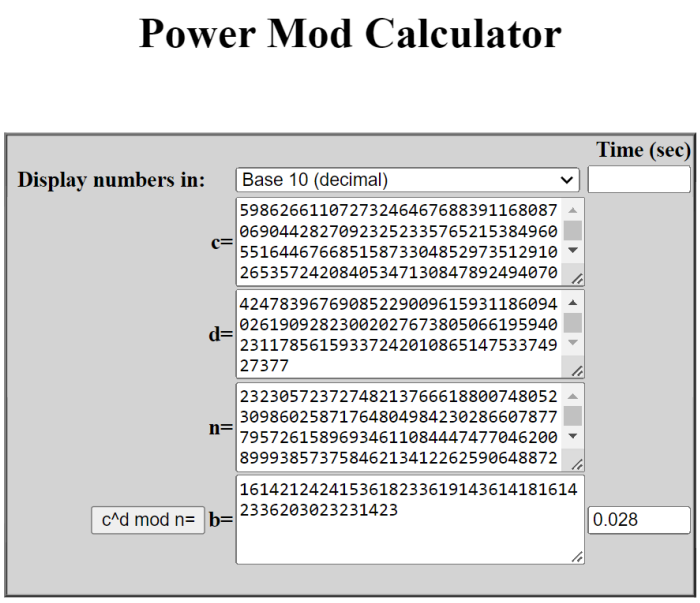
Stap 3: De code decoderen

Nu we alle berekeningen achter de rug hebben, is het tijd om de code terug om te zetten naar zijn oorspronkelijke vorm. Hiervoor kunnen we gebruikmaken van de formule die we daarnet hebben opgesteld.[24]

$$b = c^d \bmod n$$

Hierbij is *b* de boodschap die we proberen te achterhalen en *c* is de versleutelde code. Omdat *d* een redelijk groot getal is en omdat het voor gewone rekentoestellen niet mogelijk is om machten van deze grootte te berekenen, gebruiken we wederom een online rekentoestel die deze formule voor ons kan berekenen.[25] Als we alle gegevens juist invoeren komen we het volgende uit voor *b*:[26]

b = 161421242415361823361914361418161423362030232314 23



Figuur 4: Berekening van b

1. Cijfers omzetten naar letters

a	10	f	15	k	20	p	25	u	30	z	35
b	11	g	16	l	21	q	26	v	31	spatie	36
c	12	h	17	m	22	r	27	w	32		
d	13	i	18	n	23	s	28	x	33		
e	14	j	19	o	24	t	29	y	34		

Figuur 5: Het aangepaste alfabet

Als allerlaatste stap moeten we enkel nog de cijfers omzetten naar letters zodat we de boodschap kunnen lezen. Hiervoor hebben we een aangepast alfabet gekregen waarbij elke letter van het alfabet gelinkt werd aan een bepaald getal.

Na dit simpele opzoekwerk komen we het volgende uit als boodschap: “Geloof in je eigen kunnen”. Bij deze is de code gekraakt.

[24] Zie supra
[25] http://home.sandiego.edu/~cparker/old_classes/crypto_su17/powermod.html
[26] Aangezien de notatie op het online rekentoestel niet overeenkwam met de notatie die we in deze paper hanteren, hebben we deze aangepast op de schermopname.

Besluit

De ontcijferde code luidt: “geloof in je eigen kunnen”. Dit kwamen we te weten door ons te verdiepen in het modulorekenen, de kleine stelling van Fermat, de Eulerfunctie, de stelling van Euler en tenslotte RSA. Het viel ons op dat we de code zelf relatief snel hebben kunnen ontcijferen, maar dat vooral het uitleggen hoe je dit moest doen in dit werk veel tijd in beslag nam.

Dat was voor ons dan ook het leuke aan het onderzoek, we mochten wiskunde gaan bekijken vanuit een andere invalshoek. We ontdekten dingen waar we nog niks van kenden, een volledig nieuwe manier van werken! Onze nieuwsgierigheid was een grote drijfveer in dit onderzoek. We wilden zeer graag weten wat deze mysterieuze getallen nu precies wilden zeggen. De uiteindelijke boodschap: ‘geloof in je eigen kunnen’ heeft ons zeker wel boost gegeven in dit onderzoek.

Het minder leuke aspect aan het onderzoek was voor ons het neerschrijven van onze bevindingen. Dit omdat het lang duurde, en het schrijven van wiskunde een hoge concentratie vergt. Als die er niet is, zullen er hoogstwaarschijnlijk notatiefouten, rekenfouten, enzovoort insluipen.

Bibliografie

Bosma, W., & van de Craats, J. (01.06.1998). *Hoe werkt RSA?*. [14.03.2023, NEMO Kennislink: https://www.nemokennislink.nl/publicaties/hoe-werkt-rsa/].

De Naeghel, K. (21.05.2007). *RSA codering*. [16.03.2023, Koen De Naeghel: https://koendenaeghel.webs.com/RSA.pdf].

Feng, J., & Tsehaie, N. (Z.d.). *Is RSA-cryptografie nu veilig genoeg en wat betekent dit voor de toekomst van digitale beveiliging?*. [15.03.2023, Marnixreunisten: https://www.marnixreunisten.nl/wp-content/uploads/2010/06/rsa-pws.pdf].

Gupta, M. (23.01.2023). *RSA Algorithm in Cryptography*. [15.03.2023, GeeksforGeeks: https://www.geeksforgeeks.org/rsa-algorithm-cryptography/].

Hofstede, H. (Z.d.). *De kleine stelling van Fermat*. [16.03.2023, H.Hofstede: https://www.hhofstede.nl/modules/fermatklein.htm].

Hofstede, H. (Z.d.). *Modulo-rekenen*. [15.03.2023, H.Hofstede: https://www.hhofstede.nl/modules/modulo.htm].

Timmer, M. (17.12.2000). *Cryptografie-uitleg aan de hand van RSA*. [15.03.2023, Tweakers: https://tweakers.net/reviews/189/3/cryptografie-uitleg-aan-de-hand-van-rsa-algoritmes.html].

Vanhoorn, M. (Z.d.). *De kleine stelling van Fermat*. [15.03.2023, Rijksuniversiteit Groningen: https://www.math.rug.nl/~broer/pdf/ksFermat.pdf].

Van Miltenburg, O. (19.03.2021). *Waar wordt RSA gebruikt en wie is Schnorr?*. [15.03.2023, Tweakers: https://tweakers.net/reviews/8834/2/de-vernietiging-van-rsa-encryptie-met-priemgetallen-nog-niet-ten-eind-waar-wordt-rsa-gebruikt-en-wie-is-schnorr.html].

Z.a. (Z.d.). *Modulo rekenen*. [17.03.2023, Math4All: https://www.math4all.nl/venster/bekijk/modulo-rekenen/13/1].

Z.a. (Z.d.). *Security - deel 3*. [16.03.2023, Edictum: https://www.edictum.nl/lesson/security/3#].