Cryptografie met behulp van elliptische krommen

Bachelorscriptie Wiskunde

Erik van der Kouwe
Studentnummer 1397273
E-mail: erik@erisma.nl

Vrije Universiteit Amsterdam
Faculteit Exacte Wetenschappen
Afdeling Wiskunde en Informatica

Onder begeleiding van
Marcel van de Vel

16 juli 2006

Voorwoord

In 2003 heb ik gekozen voor twee studies uit dezelfde familie: wiskunde en informatica. Het onderwerp van deze scriptie ligt op het snijvlak van deze studies. Interesse in beide vakgebieden was één factor die bijdroeg aan de keuze van het onderwerp. Een andere is het feit dat ik voor werk regelmatig technieken uit de cryptografie gebruik. Voor mijn eigen onderneming ("Erisma Advies en Automatisering") maak ik maatwerk software voor klanten. Hierbij is regelmatig behoefte aan cryptografie, bijvoorbeeld voor bescherming tegen reverse engineering en om integriteit van gegevens te garanderen. Ik kende hierdoor sommige algoritmen (zoals het later genoemde RSA) uit de praktijk, maar andere (zoals het onderwerp van deze scriptie) kende ik enkel van naam.

Aangezien dit een bachelorstudie voor wiskunde is heb ik geprobeerd de algebraïsche achtergrond van het onderwerp te benadrukken. Enige achtergrondinformatie betreffende cryptografie is echter noodzakelijk. Ik wil met de scriptie graag bijdragen in het inzicht dat abstracte wiskunde, zoals algebra, nuttige toepassingen kan hebben binnen de informatica.

De informatie over cryptografie is vooral gebaseerd op Menezes, Van Oorschot en Vanstone (1996), aangevuld door Tanenbaum (2003). Merk wel op dat gebruikte letters en notaties hierbij af en toe zijn aangepast om een consistente notatie te krijgen en hergebruik van letters te voorkomen.

De basisinformatie over elliptische krommen en de meetkundige representatie daarvan zijn voornamelijk afkomstig van MathWorld en Wikipedia (zie bronvermelding). Alle definities hierin komen overeen met die van de meer conventionele bronnen, zoals Koblitz (1987), maar bieden een betere basis voor een heldere uitleg aan een algemeen publiek.

Mijn dank gaat uit naar mijn scriptiebegeleider Marcel van de Vel die enthousiast reageerde op mijn voorstel voor het onderwerp van deze scriptie en goede suggesties leverde voor verbeteringen. Verder zou ik graag mijn klanten bedanken waar ik nu al circa 5 jaar veel ervaring op kan doen en waar ik mijn kennis toe kan passen in de praktijk. Tenslotte dank ik mijn vader voor het nalezen van deze scriptie.

Erik van der Kouwe

Inhoudsopgave

1 Inleiding

Deze scriptie gaat over de toepassing van elliptische krommen voor cryptografie. We beginnen met een inleiding over cryptografie in hoofdstuk 2. We zullen hier de te bereiken doelen en de gebruikte terminologie bespreken. In hoofdstuk 3 geven we een aantal voorbeelden van encryptiemethoden.

In hoofdstuk 4 definiëren we elliptische krommen in het platte vlak en de bijbehorende groepsstructuur. Vervolgens veralgemeniseren we deze definities in hoofdstuk 5 zodat we ook over elliptische krommen over eindige lichamen kunnen spreken.

Wanneer we deze structuur hebben gedefinieerd, kunnen we enkele vormen van elliptische kromme cryptografie definiëren. We zullen dit doen in hoofdstuk 6. In hoofdstuk 7 beschouwen we de veiligheid van deze methoden en koppelen we deze aan het onopgeloste “discrete logaritme probleem”.

In hoofdstuk 8 vergelijken we cryptografie met behulp van elliptische krommen met andere vormen van cryptografie. Tenslotte trekken we in hoofdstuk 9 een conclusie over het nut van elliptische krommen binnen de cryptografie.

2 Cryptografie

2.1 Inleiding

Cryptografie is een vakgebied dat probeert de inhoud van berichten te beschermen tegen buitenstaanders. Hierbij wordt over het algemeen gebruik gemaakt van wiskundige methoden.

Van oorsprong werd cryptografie vooral voor militaire doeleinden gebruikt. Het is in dit geval van groot belang dat de tegenstander berichten die hij in handen krijgt, niet kan lezen. De verzender verstuurt daarom een onleesbaar bericht dat de ontvanger kan gebruiken om de oorspronkelijke informatie te reconstrueren. Het is van belang dat de ontvanger precies weet hoe hij dit moet doen. Nog belangrijker is dat de tegenstander deze informatie niet heeft. Dit aspect van cryptografie heet encryptie en is de meest voor de hand liggende vorm van bescherming. Een andere mogelijke vorm van bescherming van een bericht is bijvoorbeeld het beschermen tegen wijzigingen door derden.

Voorbeeld

Een vroeg gebruik van encryptie was de communicatie van Julius Caesar met zijn generaals. Als hij een vertrouwelijk bericht verstuurde, dan schoof hij elke letter drie posities op in het alfabet. Als hij bijvoorbeeld een “A” wilde oversturen, dan schreef hij in plaats daarvan “D”.

We zullen een voorbeeld bekijken. Stel dat Caesar het volgende bericht had willen versturen:

MARCVS ET CORNELIA IN HORTO AMBVLANT

We noemen dit de “plaintext”. Caesar wil nu de onleesbare “cyphertext” bepalen, zodat hij die naar zijn bevelhebbers kan verzenden. Hij maakt een tabel die aangeeft welke letter in de plaintext bij welke in de cyphertext hoort:

Plaintext	A	B	C	D	E	F	G	H	I	K	L	M	N	O	P	Q	R	S	T	V	X	Y	Z
Cyphertext	D	E	F	G	H	I	K	L	M	N	O	P	Q	R	S	T	V	X	Y	Z	A	B	C

Merk hierbij op dat de Romeinen de letters “J”, “U” en “W” niet kenden. Letter voor letter zoekt hij op in de bovenste rij en vervangt hij door de letter die eronder staat. Hij krijgt het volgende resultaat:

PDVFZX HY FRVQHOMD MQ LRVYR DPEZODQY

De ontvanger vindt het oorspronkelijke bericht door de letters in de onderste rij op te zoeken en te vervangen door die in de bovenste rij. Een vijand die het bericht in handen krijgt zal echter geen idee hebben wat er staat, zelfs als hij Latijn kan lezen. Hij kent namelijk niet het systeem om het oorspronkelijke bericht te reconstrueren.

Huidig gebruik

De aanpak die Caesar gebruikte, is tegenwoordig niet meer voldoende om gegevens goed te beveiligen. Omdat de methode algemeen bekend is, ligt het voor de hand deze te proberen als we een versleuteld bericht van de tegenstander ontvangen. Ook eenvoudige variaties erop zijn eenvoudig te breken. Men zou bijvoorbeeld een ander aantal posities kunnen verschuiven of zelfs het alfabet op een geheel andere wijze permuteren. We kunnen in het eerste geval alle 26

mogelijkheden doorlopen en kijken in welk geval we een leesbaar bericht krijgen. In het tweede geval zouden we (voor langere berichten) de code met statistische analyse kunnen breken. We kijken dan naar de frequentie waarmee letters voorkomen. Het ligt voor een Nederlandse tekst bijvoorbeeld voor de hand dat de meest voorkomende letter overeenkomt met de letter “e”, aangezien deze gemiddeld het meest voorkomt in Nederlandstalige tekst.

Om betrouwbare versleuteling te krijgen, moeten we meer doen dan enkel letters vervangen. Dit is handmatig niet meer te doen. Gebruik van een computer ligt dan voor de hand. Een mooi voorbeeld is, dat de eerste elektronische digitale computer (de Colossus) zelfs speciaal was ontwikkeld voor een cryptografische toepassing. In de Tweede Wereldoorlog gebruikten de Duitsers apparaten die ingetypte berichten versleuteld overstuurden en de Britten konden met behulp van de Colossus aardig wat van de verzonden informatie achterhalen.

Tegenwoordig wordt encryptie praktisch altijd door computers uitgevoerd. Het is niet meer een exclusief militaire techniek, maar wordt ook op grote schaal voor burgerdoeleinden gebruikt. Vooral op het internet is het vaak van belang dat gegevens niet door derden gelezen kunnen worden. Berichten worden namelijk stap voor stap doorgestuurd van verzender naar ontvanger, waarbij elke tussenschakel het bericht kan lezen. Voor gevoelige gegevens als creditcardnummers en vertrouwelijke e-mails is dit niet acceptabel.

2.2 Terminologie encryptie

Om dieper op encryptie in te kunnen gaan, zullen we eerst de relevante terminologie introduceren. We zullen er telkens vanuit gaan dat Alice een bericht naar Bob wil versturen. Een derde persoon, Trudy, luistert echter mee en kan alle communicatie tussen Alice en Bob lezen. Alice en Bob willen niet dat Trudy in staat is de verzonden boodschap te achterhalen.

We introduceren eerst een aantal verzamelingen:

- $\mathcal{P}$, de verzameling van mogelijke plaintexts. Elementen hiervan zijn berichten die Alice aan Bob zou kunnen willen versturen.
- $\mathcal{C}$, de verzameling van mogelijke cyphertexts. Elementen hiervan zijn de onleesbare berichten die verstuurd worden.
- $\mathcal{K}_e$ en $\mathcal{K}_d$, respectievelijk de verzamelingen van encryptie- en decryptiesleutels. Sleutels maken het mogelijk de encryptie- en decryptieprocedures te parametriseren.

Aangezien computers niet met overaftelbare verzamelingen kunnen werken, eisen we dat deze verzamelingen aftelbaar zijn. In de praktijk worden elementen vaak gerepresenteerd als eindige rijtjes over het alfabet $\{0,1\}$. Deze rijtjes nullen en enen zullen we verder bitstrings noemen. We kunnen bijecties vinden met alle andere aftelbaar oneindige verzamelingen, dus deze representatie beperkt de mogelijkheden niet.

We hebben verder de volgende functies:

- Encryptiefuncties $E_e : \mathcal{P} \to \mathcal{C}$ voor $e \in \mathcal{K}_e$. Deze functies beelden een plaintext af op de bijbehorende cyphertext. De gebruikte functie is afhankelijk van de gekozen sleutel. Omdat Bob de cyphertext weer om wil kunnen zetten in leesbare plaintext, moet de functie inverteerbaar zijn. Het is acceptabel dat het beeld stochastisch is, zolang geen enkele cyphertext het beeld van meerdere plaintexts is. Merk op dat E_e in dit geval niet echt een

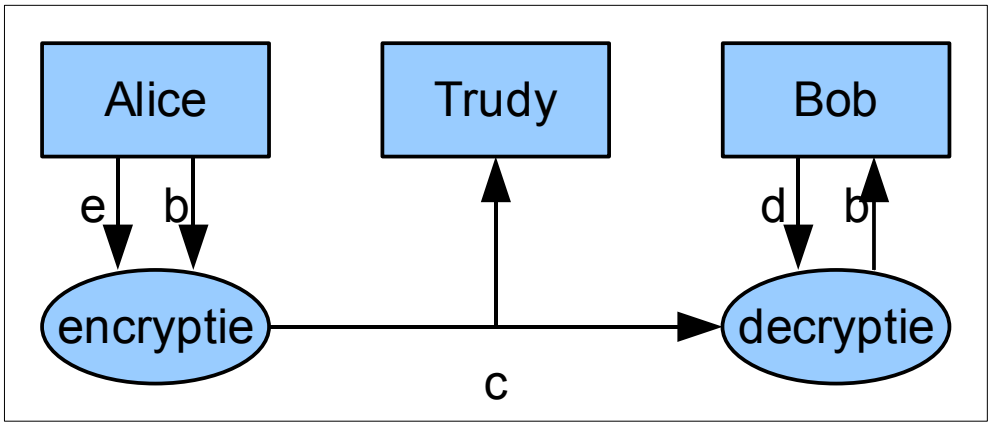
functie meer is

- Decryptiefuncties $D_d : \mathcal{C} \to \mathcal{P}$ met $d \in \mathcal{K}_d$. Deze beeldt een cyphertext af op de plaintext die hem voortgebracht heeft. De gebruikte functie hangt af van de decryptiesleutel d.

We eisen dat er voor elke encryptiesleutel $e \in \mathcal{K}_e$ een decryptiesleutel $d \in \mathcal{K}_d$ bestaat zodanig dat E_e en D_d elkaars inversen zijn. Omgekeerd dient ook elke $d \in \mathcal{K}_d$ een bijbehorende $e \in \mathcal{K}_e$ te hebben. Het kan echter wel zo zijn dat voor een zekere encryptiesleutel meerdere geldige decryptiesleutels bestaan en omgekeerd. Dit kan geen kwaad, zolang men er maar rekening mee houdt dat meerdere sleutels dezelfde encryptie- of decryptiefunctie leveren.

We gaan ervan uit dat Alice de waarde van e en Bob een bijbehorende waarde van d kent. Alice wil het bericht $b \in \mathcal{P}$ aan Bob doorgeven. Ze verstuurt dan $c := E_e(b)$ naar Bob. Hij berekent $D_d(c) = b$ en kan het bericht lezen. Trudy kent d niet en weet dus niet welke decryptiefunctie ze toe moet passen om het bericht te vinden.

De encryptieprocedure is weergegeven in afbeelding 1. De ellipsen "encryptie" en "decryptie" passen respectievelijk de functies E_e en D_d toe op de invoer en sturen het resultaat door.



Afbeelding 1: Encryptieprocedure

We gaan ervan uit dat $\mathcal{P}$, $\mathcal{C}$, $\mathcal{K}$, $(E_e)_{e \in \mathcal{K}_e}$ en $(D_d)_{d \in \mathcal{K}_e}$ algemeen bekend zijn. Verder kan Trudy ook c achterhalen door de communicatie af te luisteren. De genoemde functies moeten zodanig gekozen worden dat het voor Trudy heel moeilijk is om informatie over b af te leiden uit $E_e(b)$. In het bijzonder is het essentieel dat Trudy de waarde van d niet kent. We noemen d daarom een "private key". Onder voorwaarden is het wel acceptabel dat Trudy e kent. Het moet dan echter heel moeilijk zijn om de inverse van E_e te berekenen. De betekenis van "heel moeilijk" hier zullen we behandelen wanneer we het over de veiligheid van encryptie met elliptische krommen hebben. Het moet ook heel moeilijk zijn om d uit e af te leiden. We noemen e onder deze voorwaarden de "public key". Als niet aan de voorwaarden voldaan is, dan is ook e een private key.

Toepassing op voorbeeld

In ons eerdere voorbeeld waren $\mathcal{P}$ en $\mathcal{C}$ de verzamelingen van eindige strings over het Romeinse alfabet. We nemen nu als sleutel het aantal posities dat letters verschoven worden. Er geldt dan $\mathcal{K}_e = \mathcal{K}_d = \mathbb{Z}$. De functie $E_e(b)$ beeldt elke letter in de string b af op de letter die (cyclisch) e plaatsen verderop in het alfabet ligt. $D_d(c)$ beeldt elke letter in de string c af op de letter die

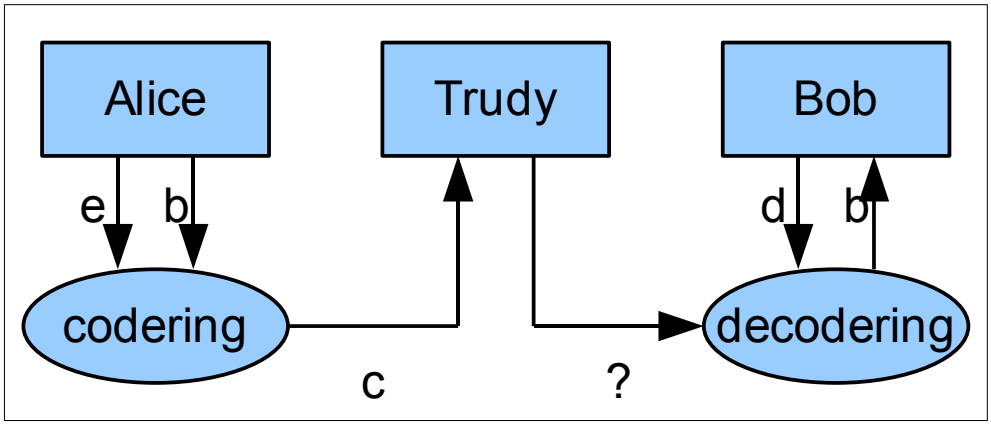
(cyclisch) d plaatsen eerder in het alfabet ligt.

Merk op dat $e=0$ een erg ongelukkige keuze voor de sleutel zou zijn. In dit geval is de cyphertext namelijk gelijk aan de plaintext. Verder zien we dat sleutels die een veelvoud van 23 verschillen, de zelfde functies E_e en D_d leveren. We zouden de sleutelverzamelingen dus ook kunnen beperken tot $\mathcal{K}_e=\mathcal{K}_d=\{0,\ldots,22\}$.

We zien dat E_e en D_d elkaars inversen zijn dan en slechts dan wanneer $e\equiv d\pmod{23}$. We hebben het in dit geval dus zeker niet over public key encryptie. Verder kan Trudy simpelweg alle 23 echt verschillende waarden voor d uitproberen en kijken in welk geval ze een leesbaar bericht krijgt. Caesar's encryptie was dus (naar huidige maatstaven) niet erg sterk.

2.3 Verdere doelen cryptografie

We hebben het nu gehad over encryptie. Dit is het meest in het oog springende gebied waar de cryptografie zich mee bezig houdt, maar cryptografie houdt meer in dan encryptie alleen. Om te zien wat er verder nodig is, passen we de situatie iets aan. Trudy is nu ook in staat om zelf berichten te verzenden en om berichten te veranderen. De nieuwe situatie is weergegeven in afbeelding 2.



Afbeelding 2: Aangepaste situatie

Het is denkbaar dat op Bob op een zeker moment ook reacties naar Alice moet kunnen versturen. Dit gaat volgens hetzelfde schema, waarbij de rollen van Alice en Bob omgewisseld zijn.

We hebben de algemenere termen “codering” en “decodering” gebruikt in plaats van “encryptie” en “decryptie” om ook problemen op kunnen te lossen waar Trudy het bericht wel mag lezen. Soms is het voor Bob namelijk vooral belangrijk dat de informatie daadwerkelijk van Alice komt en niet van Trudy. Het kan bijvoorbeeld gaan om een publieke mededeling die ook bestemd is voor Coby, Dirk, Ellen, Frank, Gerda en Hans. Het dan onpraktisch (en zinloos) om het bericht voor iedereen met zijn eigen publieke sleutel te versleutelen. Belangrijker is dat men zekerheid heeft over de afzender.

We kunnen de uiteindelijke doelen wat algemener formuleren. Volgens Menezes, Van Oorschot en Vanstone (1996) probeert men met cryptografie probeert de volgende doelen te bereiken:

- Authenticatie;
- Integriteit;
- Confidentialiteit;
- Niet-afwijzing (“non-repudiation”).

We zullen deze doelen behandelen in de context van de geschetste situatie.

Authenticatie

Authenticatie houdt in dat iemand kan bewijzen wie hij of zij is. Alice kan dit doen door te bewijzen dat ze over informatie beschikt waarvan iedereen kan weten dat alleen zij die informatie heeft. Deze informatie zelf opsturen is niet effectief, want dan kan ook de ontvanger Bob zich voortaan voordoen als Alice.

We gaan ervan uit dat Alice een public key e en een private key d heeft. Ze houdt haar private key geheim, maar maakt haar public key algemeen bekend. Een (te) simpel voorbeeld van authenticatie is dan het volgende:

- Bob kiest een willekeurige plaintext $b \in \mathcal{P}$.
- Bob berekent $c := E_e(b)$ en stuurt deze naar Alice.
- Alice berekent $D_d(c) = b$ en stuurt het resultaat naar Bob.
- Bob controleert of het antwoord gelijk is aan b, want alleen Alice kan b berekenen uit c.

Voor dit authenticatieschema hebben we enkel public key encryptie nodig. De gebruikte methode kunnen we vooraf kiezen. Mogelijkheden zijn de methoden RSA en ECIES, die we later tegen zullen komen.

Deze simpele methode is enkel ter illustratie van de aanpak bedoeld en is niet direct bruikbaar in de praktijk. Een risico is bijvoorbeeld dat Trudy aan Alice vraagt zich te identificeren door b uit c te berekenen. Ze kan b vervolgens doorgeven aan Bob. Dit soort problemen zijn op te lossen met een ingewikkelder aanpak, maar we zullen hier niet dieper op ingaan.

Een probleem is dat Bob met zekerheid moet weten dat de public key inderdaad van Alice is. Een mogelijkheid is dat een betrouwbare derde partij deze informatie controleert. We zullen deze de certification autority (afgekort CA) noemen. Deze certification autority kan een certificaat afgeven met informatie over Alice en haar public key. Als dit certificaat bewijs bevat dat het is goedgekeurd door de CA, dan kan Alice het aan Bob doorgeven en daarmee bewijs leveren over haar public key is. Om te bewijzen dat het certificaat daadwerkelijk van de CA komt zullen we een digitale handtekening gebruiken.

Integriteit

Integriteit houdt in, dat de ontvangen gegevens overeenkomen met wat er verzonden is. Digitale handtekeningen kunnen gebruikt worden voor garantie van de integriteit van een bericht. Er zijn echter ook andere methoden om integriteit te garanderen. We zullen later de encryptiemethode ECIES behandelen, waar integriteitscontrole ingebouwd zit.

Confidentialiteit

Confidentialiteit houdt in dat Trudy een bericht verstuurd van Alice naar Bob niet kan lezen. Dit is het aspect waarvoor we encryptie gebruiken.

Niet-afwijzing

Niet-afwijzing ("non-repudiation") houdt in dat Bob tegenover derden kan bewijzen dat een ontvangen bericht daadwerkelijk van Alice komt. Doel is in dit geval niet het beveiligen van gegevens tegen Trudy, maar het vastleggen van wat men zou kunnen zien als contracten. De digitale handtekening is een geschikte methode om dit doel te bereiken.

2.4 Terminologie digitale handtekeningen

We hebben gezien dat, naast encryptie, ook digitale handtekeningen nuttig zijn binnen de cryptografie. Een digitale handtekening bewijst dat een bericht ongewijzigd afkomstig is van degene die het ondertekend heeft. Plaatsen van een digitale handtekening vereist kennis van een private key, terwijl het voor verificatie van een handtekening voldoende moet zijn over de bijbehorende public key te beschikken.

We schrijven nu $\mathcal{S}$ voor de verzameling van digitale handtekeningen en, zoals eerder, $\mathcal{P}$ voor de verzameling van mogelijke berichten. We gebruiken een functie $S_d : \mathcal{P} \to \mathcal{S}$ met $d \in \mathcal{K}_d$ om de digitale handtekening voor een bericht te bepalen. Hierbij is d de private key. De functie $C_e : \mathcal{P} \times \mathcal{S} \to \{0,1\}$ met $e \in \mathcal{K}_e$ bepaalt of de gegeven handtekening bij het bericht hoort. De uitkomst is 1 als de handtekening klopt en 0 als de handtekening onjuist is.

2.5 Cryptografische primitieven

Er zijn een aantal soorten functies die we regelmatig tegenkomen als bouwstenen van encryptie en digitale handtekeningen.

Cryptografische hash

Een cryptografische hash functie is een functie $h : \{0,1\}^* \to \{0,1\}^n$ voor zekere $n \in \mathbb{N}$. Deze functie beeldt dus bitstrings van willekeurige lengte af op bitstrings van vaste lengte. Van belang hierbij is, dat het heel moeilijk is om uit het beeld van een bitstring onder een hash functie informatie over die bitstring af te leiden. In het bijzonder mag het niet mogelijk zijn snel een ander bericht te vinden dat op dezelfde hash afgebeeld wordt. Verder is het wenselijk dat de invoer zo uniform mogelijk over het bereik verspreid wordt.

Voorbeelden van bekende hash functies zijn MD5 (hiervoor geldt $n = 128$), SHA-1 ($n = 160$) en SHA-2 (hiervan zijn meerdere varianten, waarbij $n \in \{224, 256, 384, 512\}$). Deze functies gebruiken de volgende globale aanpak:

- De invoer b wordt opgedeeld in blokken $b_1, \ldots, b_m$. De blokken zijn bitstrings met vaste lengte $p \in \mathbb{N}$. Er worden bits toegevoegd om de invoer in een geheel aantal blokken te laten passen.
- Per blok b_k wordt een tussenresultaat t_k berekend. Zo'n tussenresultaat is een bitstring met vaste lengte $q \in \mathbb{N}$. Er wordt een "compressiefunctie" $f : \{0,1\}^q \times \{0,1\}^p \to \{0,1\}^q$ gebruikt. Het tussenresultaat wordt recursief bepaald door de formule $t_k := f(t_{k-1}, b_k)$, waarbij t_0 een vaste waarde heeft. De waarde van t_0 wordt de "initialisatievector" genoemd.

- Het eindresultaat wordt met een "finalisatiefunctie" $g:\{0,1\}^q \to \{0,1\}^n$ uit het laatste tussenresultaat berekend: $g(t_m)$.

De keuze van n, p, q, t_0, f en g is bepalend voor de kwaliteit van de hash functie. Vooral de keuze van de compressiefunctie f is van belang: voor een goed resultaat moet deze een uitkomst geven die moeilijk tot de invoer terug te leiden is en die zo uniform mogelijk verdeeld is. Voor de genoemde algoritmen is de compressiefunctie een ingewikkelde samenstelling van logische bewerkingen.

Sleutelafleidingsfunctie

Een sleutelafleidingsfunctie ("key derivation function", afgekort KDF) leidt een sleutel af uit de invoer. De invoer is een bitstring van willekeurige lengte, de sleutel is een bitstring van vooraf gekozen lengte. Hierbij wordt een cryptografische hash functie gebruikt zodat de sleutel zo min mogelijk informatie levert over de invoer.

Gegeven een hash functie $h:\{0,1\}^* \to \{0,1\}^n$ is het vrij eenvoudig om een sleutelafleidingsfunctie te definiëren. Een voorbeeld is de Ansi X9.63 functie. De invoer is een bitstring $w \in \{0,1\}^m$ met lengte $m \in \mathbb{N}$. We definiëren $H_k := h(w \| k)$ voor $k \in \{1, \ldots, 2^{32}-1\}$, waarbij "$\|$" de bitstringbewerking "concatenatie" aanduidt. Deze bewerking voegt twee bitstrings samen door ze achter elkaar te plakken. We vatten k op als bitstring door deze binair te schrijven met 32 cijfers (dit verklaart de bovengrens $2^{32}-1$). We kunnen nu een sleutel van lengte $p \in \mathbb{N}$ afleiden uit w door de eerste p bits uit $H_1 \| H_2 \| \ldots$ te nemen.

Message authentication code

Een "message authentication code" (afgekort MAC) is een functie die gegeven een bericht en een sleutel een code kan genereren die bewijst dat:

- De verzender van het bericht de sleutel kent.
- Het bericht sinds het aanmaken van de code niet is veranderd.

Het bericht en de sleutel zijn hierbij bitstrings van willekeurige lengte. De code is zo geconstrueerd dan deze geen informatie geeft over de sleutel, zelfs als het bericht bekend is. Hiervoor wordt een hash functie gebruikt. Een ontvanger die het bericht en de sleutel kent, kan de MAC controleren door deze zelf ook te berekenen. Als de twee codes overeenstemmen, is de code inderdaad gegenereerd door iemand de de sleutel kent en is het bericht sindsdien niet gewijzigd.

3 Voorbeelden encryptie

Alvorens in te gaan op encryptie met behulp van elliptische krommen, zullen we eerst enkele andere methoden bespreken. We zullen deze later vergelijken met elliptische kromme encryptie.

3.1 One time pad

De one time pad is een eenvoudig encryptiealgoritme waarvoor we een interessante eigenschap kunnen bewijzen: zolang men de sleutel niet kent, geeft een cyphertext geen enkele informatie over de bijbehorende plaintext.

Bij deze methode is de encryptiesleutel *e* altijd gelijk aan de bijbehorende decryptiesleutel *d*. We noemen zo'n algoritme symmetrisch. Omdat de sleutels gelijk zijn, is geen van beide sleutels een public key. Symmetrische algoritmen leveren dus altijd private key encryptie.

De encryptie- en decryptiefuncties zijn hetzelfde: beiden voeren een “bitwise xor” uit van de invoer met de sleutel. Dit houdt in dat invoer en sleutel beide bitstrings zijn van gelijke lengte en dat per bit de exclusieve of (“xor”) operatie $\oplus : \{0,1\}^2 \to \{0,1\}$ wordt uitgevoerd. De bewerking gaat volgens deze tabel:

$\oplus$	0	1
0	0	1
1	1	0

We zien direct dat de waarde van een enkel bit van de cyphertext nu niets zegt over het overeenkomstige bit in de plaintext, tenzij we ook informatie over de sleutel hebben. Beide mogelijkheden zijn namelijk even waarschijnlijk. Als de sleutel volstrekt willekeurig is, dan is er sprake van perfecte encryptie.

Tegenover deze bewijsbaar sterke encryptie staan enkele belangrijke nadelen:

- De sleutel is even lang als het bericht. Deze lange sleutel moet van tevoren via een veilig kanaal afgesproken zijn.
- De sleutel mag niet hergebruikt worden, aangezien de perfecte encryptie dan verloren gaat. De bitwise xor van twee berichten die met dezelfde sleutel zijn verzonden, levert namelijk informatie over de berichten. Als men voor één van de berichten de plaintext kent, kan zelfs de sleutel achterhaald worden, zodat men de andere kan ontcijferen.
- Generatie van echt willekeurige bits voor de sleutel is lastig. Over het algemeen moet hiervoor speciale hardware gebruikt worden die natuurkundige processen gebruikt als bron van willekeurige gegevens. Gebruik van een algoritme voor generatie van pseudo-willekeurige getallen levert gegevens die niet voldoende willekeurig zijn om het bericht perfect te beschermen.

Deze problemen zijn fundamenteel en zouden ook van toepassing zijn op andere algoritmen die perfecte encryptie leveren.

3.2 Advanced Encryption Standard

Zoals we zagen is de one time pad vaak onpraktisch in gebruik, vooral door de sleutellengte. De

Advanced Encryption Standard (AES), ook bekend als Rijndael, is een symmetrisch algoritme met een sleutel van vaste lengte. Het heeft een wedstrijd van het Amerikaanse National Institute of Standards and Technology (NIST) gewonnen en is daarmee min of meer de standaardvorm van symmetrische encryptie geworden. Sindsdien wordt het op grote schaal gebruikt, onder andere voor beveiliging van Amerikaanse staatsgeheimen.

We zullen het algoritme niet in detail behandelen, maar kijken naar het algemene idee. Het algoritme begint met een blok plaintext dat in tien tot veertien rondes wordt omgevormd tot de bijbehorende cyphertext. Het blok is opgedeeld in bytes, getallen van 0 tot en met 255. Per ronde worden de volgende stappen doorlopen:

- De sleutel wordt met de gegevens gecombineerd met behulp van de xor operatie;
- Elk byte wordt aan de hand van een tabel vervangen;
- Een aantal bytes worden onderling verwisseld;
- Er wordt een matrixvermenigvuldiging met een vaste matrix uitgevoerd. De berekeningen gebeuren binnen een eindig lichaam: de veeltermring over $\mathbb{Z}/2\mathbb{Z}$ modulo de veelterm $X^8+X^4+X^3+X^1+X^0$, de zogenaamde $GL(256)$. We zullen de algemene eindige lichamen $GL(q)$ later ook tegenkomen.

We zien dat een aantal simpele operaties samen een sterke vorm van encryptie op kunnen leveren.

3.3 RSA

RSA is waarschijnlijk de bekendste vorm van public key encryptie. De methode is relatief er eenvoudig. We zullen de aanpak hier kort omschrijven.

Aanmaken van sleutels

Bob kiest willekeurig twee grote priemgetallen p en q en berekent $n:=pq$. Omdat Bob p en q kent, kan hij de Euler functie eenvoudig uitrekenen voor dit getal: $\phi(n)=\phi(pq)=(p-1)(q-1)$. Bob kiest een getal $e\in\{1,\ldots,\phi(n)-1\}$ dat relatief priem is ten opzichte van $\phi(n)$. Tenslotte berekent hij een d zodanig dat $de\equiv 1 \pmod{\phi(n)}$. Dit kan met het uitgebreide algoritme van Euclides. De public key is (n,e) en de private key (n,d).

Encryptie

Het bericht b wordt gerepresenteerd als een getal in $\mathbb{Z}/n\mathbb{Z}$. Alice verstuurt $c:=b^e$.

Decryptie

Het bericht b wordt achterhaald door te berekenen $c^d=(b^e)^d=b^{ed}=b^1=b$, waarbij voor de tweede gelijkheid gebruik is gemaakt van $de\equiv 1 \pmod{\phi(n)}$ en de eigenschap van de Euler functie dat $b^{\phi(n)+1}\equiv b \pmod{n}$.

RSA over cyclische groepen

Zij G een cyclische groep met generator g en orde $m:=\#G$. We weten dan dat voor een willekeurig

element $a = g^n \in G$ geldt $a^m = (g^n)^m = g^{nm} = (g^m)^n = g^n = a$. Dit is precies de eigenschap die we eerder gebruikten. We kunnen RSA dus ook toepassen op algemene cyclische groepen. We kiezen dan de sleutels $d, e \in \mathbb{Z}$ zodanig dat $de \equiv 1 \pmod{m}$. Plaintexts en cyphertexts zijn elementen van G.

3.4 ElGamal

We hebben gezien dat RSA toegepast kan worden op algemene cyclische groepen. Er zijn meer public key encryptiealgoritmen waarvoor dit mogelijk is. ElGamal is hier een voorbeeld van. We zullen dit algoritme later nogmaals tegenkomen wanneer het specifiek wordt toegepast op elliptische krommen.

Voorbereiding

Voordat Alice en Bob gaan communiceren, zullen zij eerst afspraken moeten maken over welke groep en generator zij gaan gebruiken. Ze kiezen een Abelse groep G en een element $g \in G$. We gaan ervan uit dat g en G algemeen bekend zijn. Trudy is dus ook van deze waarden op de hoogte.

Aanmaken van sleutels

Het is nu van belang dat Bob (de ontvanger) een sleutel aanmaakt. Hij kiest hiervoor een willekeurig getal $d \in \mathbb{Z}$. Dit is zijn private key. Verder berekent hij $e := g^d$. Dit is de public key. Hij maakt deze algemeen bekend.

We zien voor de decryptiesleutels $\mathcal{K}_d = \mathbb{Z}$ en voor encryptiesleutels $\mathcal{K}_e = \{ g^z \mid z \in \mathbb{Z} \}$, de cyclische groep voortgebracht door g.

Encryptie

We gaan ervan uit, dat het bericht $b \in G$ gerepresenteerd is als element van de groep. Hoe men dit efficiënt kan doen hangt af van de groep die men gebruikt. Alice kiest een willekeurig getal $d' \in \mathbb{Z}$ en verstuurt een paar van groepselementen: $c := (g^{d'}, be^{d'})$. Zowel Bob als Trudy ontvangen deze informatie.

Merk op dat de encryptiefunctie E_e hier stochastisch is. Het is van belang dat Trudy niet kan voorspellen welk willekeurig getal Alice kiest. Als ze d' wel kan voorspellen kan ze b eenvoudig uitrekenen uit het tweede punt, want ze kent ook e. We zien verder dat $\mathcal{P} = G$ en $\mathcal{C} = G^2$, dus een cyphertext neemt meer opslagruimte in dan een plaintext. Dit is nodig doordat elke plaintext vele verschillende representaties als cyphertext heeft.

Decryptie

Bob ontvangt het paar $(g^{d'}, be^{d'})$. Hij kent d en kan dus met behulp van het eerste groepselement in het paar $e^{d'}$ als volgt bepalen:

$$e^{d'} = (g^d)^{d'} = g^{dd'} = g^{d'd} = (g^{d'})^d$$

Vervolgens kan hij het bericht bepalen uit het tweede groepselement, want $b = be^{d'}(e^{d'})^{-1}$.

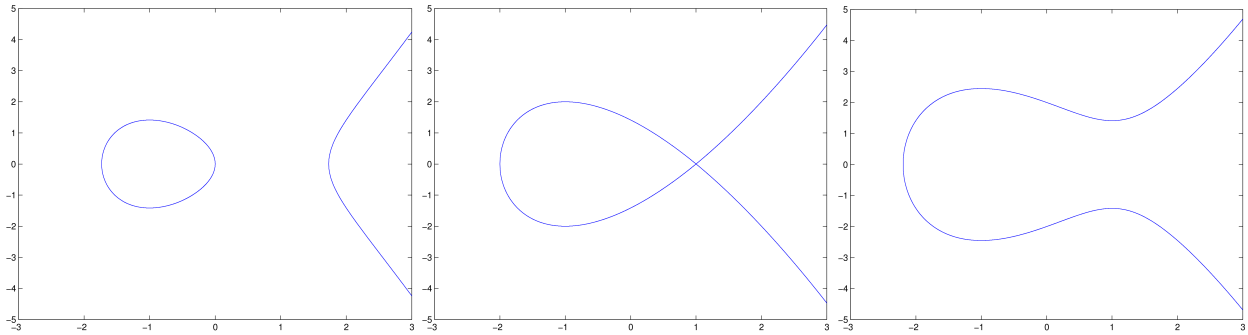
4 Elliptische krommen

4.1 Inleiding

Een elliptische kromme is een kromme in het platte vlak, die de volgende vorm heeft:

$$\left\{(x,y)\in\mathbb{R}^2 \,\middle|\, y^2=x^3+a_1x+a_2\right\}$$

Hierbij zijn $a_1, a_2\in\mathbb{R}$. Om een idee te krijgen van hoe deze krommen eruit zien, hebben we een aantal voorbeelden afgebeeld:



Kromme met a_1 = -3, a_2 = 0 *Kromme met a_1 = -3, a_2 = 2* *Kromme met a_1 = -3, a_2 = 4*

Het valt op, dat we hier met twee duidelijk verschillende gevallen en een tussenliggend randgeval te maken hebben. Deze zijn te onderscheiden naar het aantal punten op de x-as. Hier geldt de kubische vergelijking $x^3+a_1x+a_2=0$. Zo'n vergelijking heeft op grond van de hoofdstelling van de algebra drie nulpunten. Hiervan zijn er of één of drie reëel. Het is mogelijk dat meerdere nulpunten samenvallen, zodat er een nulpunt met hogere multipliciteit ontstaat. Dit is precies het geval op het moment dat drie nulpunten overgaan in één (en omgekeerd). Om te bepalen wanneer dit gebeurt, factoriseren we de veelterm:

$$x^3+a_1x+a_2=(x-r_1)(x-r_2)(x-r_3)$$

Hier zijn $r_1, r_2, r_3\in\mathbb{C}$ de nulpunten. We zoeken het geval waar twee nulpunten samenvallen, dus $r_2=r_3$. De formule wordt zo:

$$x^3+a_1x+a_2=(x-r_1)(x-r_2)(x-r_2)=x^3-(r_1+2r_2)x^2+(2r_1r_2+r_2^2)x-r_1r_2^2$$

We splitsen de vergelijking naar de machten van x en vinden:

$$\begin{aligned} -(r_1+2r_2)=0 &\quad\Rightarrow\quad r_1=-2r_2 \\ 2r_1r_2+r_2^2=a_1 &\quad\Rightarrow\quad a_1=-3r_2^2 \\ -r_1r_2^2=a_2 &\quad\Rightarrow\quad a_2=2r_2^3 \end{aligned}$$

We zien hierdoor dat het randgeval plaatsvindt wanneer $4a_1^3+27a_2^2=0$. We hebben een samenhangende kromme als $4a_1^3+27a_2^2>0$ (het geval rechts). In het andere geval valt de kromme uiteen in twee componenten, zoals links afgebeeld. We beschouwen het randgeval $4a_1^3+27a_2^2=0$ als ongewenst en zullen verder eisen dat $4a_1^3+27a_2^2\neq 0$.

Elliptische krommen staan in feite model voor alle kubische krommen. Het is namelijk mogelijk

alle krommen met de onderstaande representatie door verandering van coördinaten om te zetten in een elliptische kromme:

$$A x^3 + B x^2 y + C x y^2 + D y^3 + E x^2 + F x y + G y^2 + H x + I y + J = 0$$

Er zijn verschillende onderzoeksgebieden in de wiskunde waarin elliptische krommen interessant zijn. Topologisch kunnen we de kromme bijvoorbeeld zien als een in $\mathbb{R}^2$ ingebedde één-dimensionale variëteit mits $4a_1^3 + 27a_2^2 \neq 0$. We kunnen de kromme echter ook als algebraïsch object zien. Het is namelijk mogelijk een groepsstructuur definiëren over de elliptische kromme. Deze groepsstructuur zullen we gebruiken voor encryptie. Hierbij zullen we ons niet meer beperken tot elliptische krommen over $\mathbb{R}$, maar zullen we ook elliptische krommen over andere lichamen beschouwen.

4.2 Geschiedenis

Uit de formule horende bij een elliptische kromme is niet direct duidelijk wat deze met een ellips te maken heeft. De naamgeving is te danken aan de geschiedenis van de kromme.

Deze geschiedenis begint bij de elliptische integralen. Dit zijn integralen die gebruikt kunnen worden om de booglengte van een stuk ellips te bepalen. Deze aanpak lijkt sterk op de berekening van de inverse sinus en inverse cosinus voor een cirkel, aangezien ook deze een stuk booglengte berekenen. Als we deze parallel verder trekken, kunnen we proberen een elliptische sinus en cosinus te vinden. Dit is dan de inverse van de elliptische integraal. Aan dit onderwerp is gewerkt door Jacobi en Weierstraß.

Het werk van laatstgenoemde leidde tot de Weierstraß elliptische functie $\wp$. Men kan een uitdrukking vinden voor deze functie als differentiaalvergelijking:

$$(\wp'(z))^2 = 4\wp^3(z) + g_2\wp(z) + g_3$$

Als we de functiewaarde op de x-as plaatsen en de afgeleide op de y-as, hebben we hiermee de formule van een elliptische kromme. De factor 4 kan weggewerkt worden door een eenvoudige substitutie en aanpassen van de constanten. Dit is waar de formule voor de elliptische kromme vandaan komt; de meetkundige betekenis is achteraf pas aan de kromme gekoppeld.

Voor de Weierstraß $\wp$ functie geldt verder de volgende somformule:

$$\wp(z+y) = \frac{1}{4}\left(\frac{\wp'(z)-\wp'(y)}{\wp(z)-\wp(y)}\right)^2 - \wp(z) - \wp(y)$$

Deze zullen we verderop ook tegenkomen, namelijk als definitie van de optelling op elliptische krommen. Wederom is achteraf een meetkundige betekenis aan de formule toegekend, die we verderop tegen zullen komen.

Uiteindelijk bleek de elliptische kromme van groot nut binnen de getaltheorie en later cryptografie. De koppeling met de oorspronkelijke betekenis valt hierbij weg, want de vergelijking wordt in dit geval meestal toegepast op eindige lichamen in plaats van op $\mathbb{R}$. Dit zullen wij verderop ook doen.

4.3 Groepsstructuur

We willen een groepsstructuur op de elliptische kromme definiëren. We schrijven $a := (a_1, a_2)$ en gebruiken verder de notatie $\mathcal{E}_a$ voor de kromme $\left\{(x, y) \in \mathbb{R}^2 \middle| y^2 = x^3 + a_1 x + a_2\right\}$ plus een nieuw punt

$\mathcal{O}$, dat we het punt op oneindig zullen noemen. Dit punt zal de rol van neutraal element in onze groep vervullen.

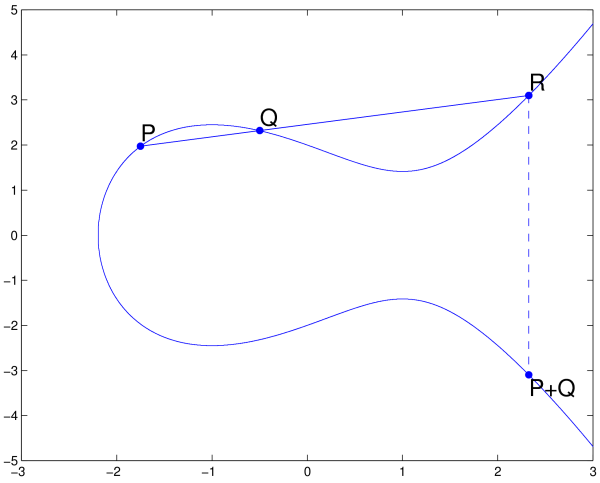
Zij nu $P, Q \in \mathcal{E}_a$. We zullen de optelling opsplitsen in verschillende gevallen en per geval definiëren. We zullen later voor elliptische krommen over algemene lichamen laten zien dat de optelling met de hier gegeven definitie inderdaad een groepsbewerking vormt.

P *of* Q *gelijk aan het punt op oneindig*

Als $P = \mathcal{O}$ of $Q = \mathcal{O}$, dan volgt de waarde van $P + Q$ direct uit de rol van het punt op oneindig als neutraal element. Als $P = \mathcal{O}$ dan definiëren we $P + Q := Q$. Als $Q = \mathcal{O}$ dan hebben we $P + Q := P$. We hebben het geval $P = Q = \mathcal{O}$ op deze wijze dubbel gedefinieerd, maar op consistente wijze.

P_x *ongelijk aan* Q_x

In andere gevallen is de definitie voor onze optelling gebaseerd op meetkunde. We kunnen dan schrijven $P = (P_x, P_y)$ en $Q = (Q_x, Q_y)$. Als nu $P_x \neq Q_x$ dan trekken we een lijn door P en Q. Deze lijn snijdt de kromme in een derde punt $R = (R_x, R_y)$. Er is altijd een unieke R die hieraan voldoet (we zullen straks zien waarom). De som van P en Q definiëren we als R gespiegeld in de y-as. Het onderstaande plaatje geeft de situatie weer:



Afbeelding 3: Optelling op de kromme

Als we de gevonden lijn schrijven als $y = c_1 x + c_2$ voor $c_1, c_2 \in \mathbb{R}$, dan vinden we $c_1 = \frac{P_y - Q_y}{P_x - Q_x}$ en $c_2 = \frac{P_x Q_y - P_y Q_x}{P_x - Q_x}$. We moeten het volgende stelsel van vergelijkingen oplossen om R te vinden:

$$R_y = c_1 R_x + c_2$$
$$R_y^2 = R_x^3 + a_1 R_x + a_2$$

We vullen de eerste vergelijking in in de tweede en we vinden

$$c_1^2 R_x^2 + 2 c_1 c_2 R_x + c_2^2 = R_x^3 + a_1 R_x + a_2 \text{ oftewel } R_x^3 - c_1^2 R_x^2 + (a_1 - 2 c_1 c_2) R_x + a_2 - c_2^2 = 0$$

Op grond van de hoofdstelling van de algebra heeft deze kubische veelterm precies drie nulpunten.

Hiervan kennen we er al twee, namelijk P_x en Q_x. Beide zijn reëel en dus moet ook het derde nulpunt reëel zijn. We concluderen dat R inderdaad altijd bestaat en uniek is.

Met behulp van veeltermstaartdeling kunnen we de ongewenste nulpunten (namelijk $R_x = P_x$ en $R_x = Q_x$) wegdelen om een formule voor R_x te krijgen. We gebruiken daarbij telkens de vergelijking van de elliptische kromme om te laten zien dat de rest nul is. Na deling door $R_x - P_x$ krijgen we:

$$R_x^2 + (P_x - c_1^2)\,R_x + P_x^2 - P_x c_1^2 - 2\,c_1 c_2 + a_1 = 0$$

Als we ook delen door $R_x - Q_x$ blijft de volgende vergelijking over:

$$R_x + P_x + Q_x - c_1^2 = 0$$

We kunnen nu een directe formule geven voor de ligging van het derde snijpunt:

$$R_x = c_1^2 - P_x - Q_x$$
$$R_y = c_1 R_x + c_2 = c_1 R_x + P_y - c_1 P_x$$

Als we dit punt spiegelen in de x-as, dan krijgen we $P + Q$. Deze formule zullen we voor algemene lichamen gebruiken als definitie, aangezien de meetkundige aanpak daar niet van toepassing is. Conclusie:

$$(P+Q)_x = c_1^2 - P_x - Q_x$$
$$(P+Q)_y = c_1 (P_x - R_x) - P_y$$

P gelijk aan Q

Als $P_x = Q_x$, dan werkt de bovengenoemde aanpak niet, want we zouden bij bepaling van c_1 door nul delen. Er zijn twee mogelijkheden:

- P is gelijk aan Q
- P en Q zijn elkaars gespiegelden in de x-as

We zullen nu het eerste geval bekijken. In dit geval tellen we het getal bij zichzelf op. Het gaat hier in feite om een verdubbeling. We schrijven daarom ook wel $P + P = 2\,P$.

We zien dit geval als limietsituatie van het naar P toeschuiven van Q. Resultaat is dat de lijn door P en Q overgaat in de raaklijn in P. De situatie is weergegeven in afbeelding 4.

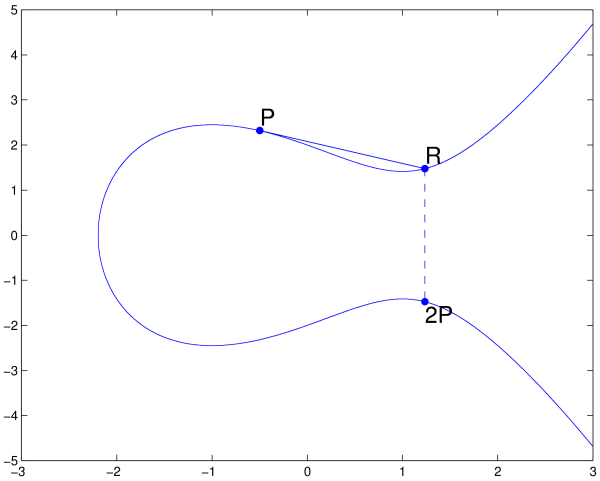
Wederom willen we een directe formule vinden voor de coördinaten van R. We bepalen eerst de formule van de lijn. Als we schrijven $y = c_1 x + c_2$, dan moet c_1 de richtingscoëfficiënt van de kromme in P zijn. We vinden:

$$\frac{dy}{dx} = \frac{3x^2 + a_1}{2\sqrt{x^3 + a_1 x + a_2}} = \frac{3x^2 + a_1}{2y} \text{ zodat } c_1 = \frac{3P_x^2 + a_1}{2P_y}$$

Merk op dat de formule niet werkt als $P_y = 0$. Dit speciale geval behandelen we verderop.

Voor de rest gaat de redenering die we gebruikten als $P_x \neq Q_x$ nog steeds op. We vinden dan ook weer dezelfde uitkomst, zij het dat c_1 nu een andere waarde heeft:

$$(P+Q)_x = c_1^2 - P_x - Q_x$$
$$(P+Q)_y = c_1 (P_x - R_x) - P_y$$



Afbeelding 4: Verdubbeling op de kromme

P en Q gespiegelden

We hebben nu nog één geval over: $P_x = Q_x$ met $P_y = -Q_y$. In dit geval zijn de punten elkaars spiegelingen in de x-as. Een lijn door beide punten is nu verticaal en snijdt nooit een derde punt. Ook het geval $P_y = 0$, dat we eerder niet konden behandelen, valt hieronder. In dit geval snijdt de raaklijn verder geen enkel punt. In al deze gevallen definiëren we $P + Q = \mathcal{O}$. De redenatie hierachter is, dat we het punt op oneindig kiezen omdat het snijpunt "in het oneindige ligt".

Deze keuze houdt in dat spiegelen in de x-as overeenkomt met het nemen van de tegengestelde. P is door deze definitie namelijk de tegengestelde van Q en omgekeerd. In het bijzonder zijn punten P met $P_y = 0$ hun eigen tegengestelde.

5 Krommen over algemene lichamen

5.1 Algemene definitie

We hebben een optelling gedefinieerd, maar nog niet aangetoond dat deze inderdaad een groepsbewerking is. We zullen dit namelijk doen voor elliptische krommen over algemene lichamen. De meetkundige betekenis van de optelling gaat hierbij verloren, maar bij juiste keuze van het lichaam blijven de formules bruikbaar.

De "juiste keuze" houdt in dit geval in dat de karakteristiek van het lichaam niet gelijk mag zijn aan twee of drie. De karakteristiek van een ring R is het kleinste gehele getal $n \in \mathbb{N}$ waarvoor $n \cdot 1 = 0$, waarbij $1 \in R$ en $0 \in R$ respectievelijk het eenheidselement en het neutraal element van de ring zijn. Merk op dat het getal n mogelijk niet in de ring zit. We passen hier ook dan niet de vermenigvuldiging in de ring toe, maar voeren herhaalde optelling uit. Als $n = 3$, dan geldt dus bijvoorbeeld $1+1+1=0$. We zullen deze notatie later nog enkele malen gebruiken. Als er geen n met deze eigenschap bestaat, dan definiëren we de karakteristiek als nul.

De restrictie op de karakteristiek is gerelateerd aan de eerdere opmerking, dat elke kubische kromme als elliptische kromme geschreven kan worden. Deze constructie kan niet overgezet worden naar lichamen met karakteristiek twee of drie, dus in dit geval is een andere definitie voor elliptische krommen nodig om de eigenschap te behouden. We zullen deze definitie geven wanneer we ingaan op elliptische krommen over eindige lichamen.

Zij K een lichaam waarvan de karakteristiek niet twee of drie is. Zij $a_1, a_2 \in K$ met $4a_1^3 + 27a_2^2 \neq 0$. Wederom passen we herhaalde optelling toe, dus $4a_1^3$ is niets meer dan een verkorte schrijfwijze voor $a_1^3 + a_1^3 + a_1^3 + a_1^3$.

We definiëren een elliptische kromme $\mathcal{E}_a[K]$ over K als volgt:

$$\mathcal{E}_a[K] := \left\{ (x, y) \in K^2 \,\middle|\, y^2 = x^3 + a_1 x + a_2 \right\} \cup \{\mathcal{O}\}$$

Hierbij is $\mathcal{O}$ een object dat niet in K^2 zit.

5.2 Groepsstructuur

We definiëren verder een groepsbewerking $+ : \mathcal{E}_a[K] \times \mathcal{E}_a[K] \to \mathcal{E}_a[K]$ op de elliptische kromme. Net als eerder splitsen we de definitie op in vier gevallen:

1. Voor alle $P \in \mathcal{E}_a[K]$ geldt $P + \mathcal{O} = \mathcal{O} + P = P$
2. Voor alle $P \in \mathcal{E}_a[K]$ geldt $(P_x, P_y) + (P_x, -P_y) = \mathcal{O}$
3. Voor alle $P, Q \in \mathcal{E}_a[K]$ waarvoor $P_x \neq Q_x$ geldt $P + Q = R$ met $R_x = \lambda^2 - P_x - Q_x$, $R_y = \lambda(P_x - R_x) - P_y$ en $\lambda = (P_y - Q_y)(P_x - Q_x)^{-1}$

4. Voor alle $P \in \mathcal{E}_a[K]$ geldt $P+P=R$ met $R_x=\lambda^2-2P_x$, $R_y=\lambda(P_x-R_x)-P_y$ en $\lambda=(3P_x^2+a_1)(2P_y)^{-1}$

We willen aantonen dat $+$ op deze wijze inderdaad een groepsbewerking is. We moeten dan aantonen dat voor alle $P, Q, R \in \mathcal{E}_a[K]$ aan de volgende voorwaarden is voldaan:

- Associativiteit: $(P+Q)+R=P+(Q+R)$
- Neutraal element: $P+\mathcal{O}=\mathcal{O}+P=P$
- Tegengestelde: er bestaat een $P' \in \mathcal{E}_a[K]$ zodanig dat $P+P'=P'+P=\mathcal{O}$.

De tweede en derde eis zijn hier triviaal, want deze zijn rechtstreeks in de definitie van de bewerking opgenomen. De tegengestelde is hierdoor de in de x-as gespiegelde. Associativiteit is hier het lastige punt.

In de formulering van associativiteit wordt de bewerking toegepast op vier verschillende paren van punten. Aangezien de bewerking vier gevallen onderscheidt, zouden er in het bewijs $4^4=256$ gevallen onderscheiden moeten worden. Op grond van symmetrieoverwegingen kan dit aantal wel wat gereduceerd worden en is het denkbaar dat alternatieve benaderingen wat gevallen elimineren, maar feit blijft dat een bewijs veel ruimte kost zonder dat het veel inzicht geeft. We zullen het hier dus achterwege laten. We verwijzen naar Charlap en Robbins (1988) voor een volledig bewijs van associativiteit van de optelling op elliptische krommen.

Het blijkt dat de gedefinieerde bewerking nog een extra eigenschap heeft. De gevonden bewerking is ook commutatief. De gevonden groepsstructuur op elliptische krommen is dus Abels.

We gaan commutativiteit na per geval. We volgen hierbij de nummering zoals eerder gehanteerd:

1. Commutativiteit staat direct in de formulering.
2. Kies $P \in \mathcal{E}_a[K]$. Toepassen van dezelfde regel op $(P_x, -P_y)$ levert

   $$(P_x, -P_y)+(P_x, --P_y)=(P_x, -P_y)+(P_x, P_y)=\mathcal{O}$$

3. Bij verwisseling van P en Q blijft λ gelijk, want

   $$(P_y-Q_y)(P_x-Q_x)^{-1}=(Q_y-P_y)(Q_x-P_x)^{-1}$$

   Het is dan direct duidelijk dat ook R_x gelijk blijft onder verwisseling van P_x en Q_x. Voor R_y schrijven we tenslotte:

   $$\begin{aligned} R_y &= \lambda(P_x-R_x)-P_y \\ &= (P_y-Q_y)(P_x-Q_x)^{-1}(P_x-R_x)-P_y \\ &= ((P_y-Q_y)(P_x-R_x)-(P_x-Q_x)P_y)(P_x-Q_x)^{-1} \\ &= (P_yQ_x-P_xQ_y+(Q_y-P_y)R_x)(P_x-Q_x)^{-1} \end{aligned}$$

   Deze uitdrukking is invariant onder verwisseling van P en Q. Als we de twee namelijk verwisselen, veranderen enkel de tekens van beide factoren gelijktijdig. Het netto effect is dat de waarde gelijk blijft.

4. Beide argumenten zijn gelijk, dus commutativiteit volgt direct.

5.3 Voorbeeld van een kromme over $\mathbb{Z}/p\mathbb{Z}$

Voor encryptie zullen we gebruik maken van krommen over eindige lichamen. Zij $p \in \mathbb{N}$ een priemgetal. Dan vormen de gehele getallen modulo p, verder genoteerd als $\mathbb{Z}/p\mathbb{Z}$, een eindig lichaam met p elementen. De karakteristiek van dit lichaam is ook p, dus we kunnen de gegeven definitie gebruiken als $p > 3$. Om een beeld te krijgen van wat een elliptische kromme over een eindig lichaam is, zullen we een voorbeeld uitwerken.

We zullen in ons voorbeeld kiezen voor $p = 5$. Dit levert de onderstaande tabellen voor optelling, tegengestelde, vermenigvuldiging en inverse. We lezen de getallen hier als klassen modulo 5.

+	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

x	-x
0	0
1	4
2	3
3	2
4	1

·	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

x	x^{-1}
1	1
2	3
3	2
4	4

We kunnen nu bijvoorbeeld een kromme kiezen door te stellen $a_1 = \overline{1}$ en $a_2 = \overline{4}$. Deze waarden zijn toegelaten, want:

$$4a_1^3 + 27a_2^2 = 4(\overline{1})^3 + 27(\overline{4})^2 = 4\cdot\overline{1} + 27\cdot\overline{1} = \overline{4} + \overline{2} = \overline{1} \neq \overline{0}$$

Per punt $(x, y) \in (\mathbb{Z}/5\mathbb{Z})^2$ kunnen we nu bepalen of deze op de kromme ligt. We doen dit door te bepalen of $y^2 = x^3 + a_1 x + a_2$. We hebben de waarde van $y^2 - x^3 + \overline{1}x + \overline{4}$ voor alle punten bepaald. De punten waarvoor hier $\overline{0}$ uitkomt, liggen op de kromme:

y\x	0	1	2	3	4
0	1	4	1	1	3
1	2	0	2	2	4
2	0	3	0	0	2
3	0	3	0	0	2
4	2	0	2	2	4

We kunnen de kromme nu expliciet als verzameling opschrijven:

$$\mathcal{E}_{(\overline{1},\overline{4})}[\mathbb{Z}/5\mathbb{Z}] := \left\{(\overline{0},\overline{2}), (\overline{0},\overline{3}), (\overline{1},\overline{1}), (\overline{1},\overline{4}), (\overline{2},\overline{2}), (\overline{2},\overline{3}), (\overline{3},\overline{2}), (\overline{3},\overline{3}), \mathcal{O}\right\}$$

We zullen ook twee voorbeelden van de optelling uitwerken: een geval van twee punten met verschillende *x* en een geval met twee identieke punten (een verdubbeling).

Zij $P = (\overline{2},\overline{3})$ en $Q = (\overline{0},\overline{2})$, zodat $P, Q \in \mathcal{E}_{(\overline{1},\overline{4})}[\mathbb{Z}/5\mathbb{Z}]$. We berekenen $P + Q$ en $2P$.

Per definitie geldt $(P_x, P_y) + (Q_x, Q_y) = (R_x, R_y)$, waarbij:

- $\lambda = (P_y - Q_y)(P_x - Q_x)^{-1} = (\overline{3} - \overline{2})(\overline{2} - \overline{0})^{-1} = \overline{3}$
- $R_x = \lambda^2 - P_x - Q_x = \overline{3}^2 - \overline{2} - \overline{0} = \overline{2}$
- $R_y = \lambda(P_x - R_x) - P_y = \overline{3}\cdot(\overline{2} - \overline{2}) - \overline{3} = \overline{2}$

We concluderen $(\overline{2},\overline{3}) + (\overline{0},\overline{2}) = (\overline{2},\overline{2})$. Zoals verwacht ligt dit punt netjes op de kromme.

Voor $2P = (P_x, P_y) + (P_x, P_y) = (R_x, R_y)$ gaat de berekening als volgt:

- $\lambda=(3P_x^2+a_1)(2P_y)^{-1}=(3\cdot\overline{2}^2+\overline{1})(2\cdot\overline{3})^{-1}=\overline{3}$
- $R_x=\lambda^2-2P_x=\overline{3}^2-2\cdot\overline{2}=\overline{0}$
- $R_y=\lambda(P_x-R_x)-P_y=\overline{3}(\overline{2}-\overline{0})-\overline{3}=\overline{3}$

Het resultaat is dus $2(\overline{2},\overline{3})=(\overline{0},\overline{3})$. Wederom ligt het punt op de kromme.

We zien dat het bepalen van sommen op elliptische krommen vrij veel rekenwerk kost. In de praktijk worden voor encryptie veel grotere eindige lichamen gebruikt, zodat deze berekeningen ook nog eens moeten gebeuren op grote getallen. Encryptie met elliptische krommen vergt hierdoor vrij veel rekenkracht.

5.4 Krommen over eindige lichamen

Voor encryptie worden elliptische krommen gebruikt over eindige lichamen. Hiervan is $\mathbb{Z}/p\mathbb{Z}$ met p priem een voorbeeld, maar het kan nuttig zijn om algemenere eindige lichamen te beschouwen. Als $p,n\in\mathbb{N}$ met p priem, dan bestaat er een uniek eindig lichaam $GF(p^n)$ met p^n elementen. Vanwege deze uniciteit is $\mathbb{Z}/p\mathbb{Z}$ isomorf met $GF(p)$. De karakteristiek van $GF(p^n)$ is p, dus als $p>3$ kunnen we deze lichamen veilig gebruiken in onze eerdere definities.

Het blijkt echter dat in praktische situaties de lichamen $GF(2^n)$ heel prettig zouden zijn om te gebruiken. Aangezien computers binair rekenen, zijn deze lichamen natuurlijker voor de computer en dus kan er efficiënt mee gerekend worden. We hebben dan definities voor de elliptische kromme en de bewerkingen daarop nodig als de karakteristiek twee is. In dit geval kunnen we een aantal termen in de algemene kubische formule niet meer wegwerken. We zullen de definities hier geven, maar gaan er inhoudelijk niet verder op in.

Zij K een lichaam met karakteristiek twee of drie. Zij $a=(a_1,a_2,a_3,a_4)\in K^4$ met $4a_1^3+27a_2^2\neq 0$. We definiëren een elliptische kromme $\mathcal{E}_a[K]$ over K als volgt:

$$\mathcal{E}_a[K]:=\left\{(x,y)\in K^2 \middle| y^2+a_3xy+a_4y=x^3+a_1x+a_2\right\}\cup\{\mathcal{O}\}$$

Hierbij is $\mathcal{O}$ een object dat niet in K^2 zit.

We definiëren verder een groepsbewerking $+:\mathcal{E}_a[K]\times\mathcal{E}_a[K]\to\mathcal{E}_a[K]$ op de elliptische kromme. We splitsen splitsen de definitie op in vier gevallen:

1. Voor alle $P\in\mathcal{E}_a[K]$ geldt $P+\mathcal{O}=\mathcal{O}+P=P$
2. Voor alle $P_x,P_y\in K$ geldt $(P_x,P_y)+(P_x,-P_y)=\mathcal{O}$
3. Voor alle $P_x,P_y,Q_x,Q_y\in K$ waarvoor $P_x\neq Q_x$ geldt $(P_x,P_y)+(Q_x,Q_y)=(R_x,R_y)$ met $R_x=\lambda^2+\lambda a_3-P_x-Q_x$, $R_y=\lambda(P_x-R_x)-P_y-a_3R_x-a_4$ en $\lambda=(P_y-Q_y)(P_x-Q_x)^{-1}$
4. Voor alle $P_x,P_y\in K$ geldt $(P_x,P_y)+(P_x,P_y)=(R_x,R_y)$ met $R_x=\lambda^2+\lambda a_3-2P_x$, $R_y=\lambda(P_x-R_x)-P_y-a_3R_x-a_4$ en $\lambda=(3P_x^2+a_1-a_3P_y)(2P_y+a_3P_x+a_4)^{-1}$

Merk op dat de definitie geheel identiek is aan die van het eerdere geval wanneer we kiezen $a_3=a_4=0$. We hoeven ons dus niet al te veel zorgen te maken over het karakteristiek van K; het enige probleem met de eerdere definitie is dus dat sommige krommen voor karakteristieken twee en

drie over het hoofd gezien zouden worden.

We zullen dit geval verder niet specifiek behandelen, maar de resultaten voor $a=(a_1, a_2)$ gaan ook op voor $a=(a_1, a_2, a_3, a_4)$.

6 Encryptie en decryptie

Er zijn verschillende manieren om encryptie uit te voeren met behulp van elliptische krommen. We zullen eerst kijken naar een aanpak van ElGamal toegepast op elliptische krommen. Daarna kijken we naar ECIES, een ingewikkelder methode die meer mogelijkheden biedt.

6.1 Aanpak van ElGamal / Koblitz

We hebben de methode van ElGamal eerder behandeld. We zullen nu specifiek ingaan op de toepassing hiervan op elliptische krommen. Deze aanpak is afkomstig van Koblitz.

Het is van belang op te merken dat we eerder de multiplicatieve notatie hebben gebruikt omdat die voor ElGamal op algemene groepen gebruikelijk is. Voor elliptische krommen wordt echter altijd de additieve notatie gebruikt. Een verder notationeel verschil is dat de generator nu G heet in plaats van g. Dit komt door de conventie om punten op de kromme met een hoofdletter te noteren, zoals we dat eerder ook al deden.

Voorbereiding

Als een elliptische kromme wordt gebruikt voor ElGamal encryptie, dan zullen Alice en Bob eerst afspraken moeten maken over de parameters van de elliptische kromme die zij gaan gebruiken. Verder moet wederom een generator gekozen worden. Al met al moeten ze de volgende parameters vooraf kiezen:

- Het lichaam waarop de elliptische kromme gedefinieerd is. We kiezen een eindig lichaam, dus $K = GF(q)$ voor $q = p^n$ met $p, n \in \mathbb{N}$ en p priem;
- De kromme die gebruikt wordt. De kromme $\mathcal{E}_a[K]$ wordt vastgelegd door a. Afhankelijk van de karakteristiek van K kiezen we $a = (a_1, a_2) \in K^2$ of $a = (a_1, a_2, a_3, a_4) \in K^4$;
- Een punt $G \in \mathcal{E}_a[K]$ op de kromme om als generator te dienen.

We gaan ervan uit dat q, a en G algemeen bekend zijn.

Aanmaken van sleutels

Net als eerder kiest Bob een willekeurig getal $d \in \mathbb{Z}$ als private key. Verder maakt hij $e := dG$ bekend als public key. Merk op dat de vermenigvuldiging bij additieve notatie een herhaalde optelling is. Deze komt overeen met de exponentiatie bij de multiplicatieve notatie, die immers een herhaalde vermenigvuldiging is.

We zien voor de decryptiesleutels $\mathcal{K}_d = \mathbb{Z}$ en voor encryptiesleutels $\mathcal{K}_e = \{ zG \mid z \in \mathbb{Z} \}$.

Encryptie

Om een bericht $b \in \mathcal{E}_a[K]$ te versturen naar Bob kiest Alice eerst een willekeurig getal $d' \in \mathbb{Z}$. Ze verstuurt een paar van punten op de kromme: $c := (d'G, b + d'e)$.

Decryptie

Net als in het algemene geval kan Bob $d'e$ bepalen uit de private key d en het eerste punt van c. Door $d'e$ van het tweede punt af te trekken achterhaalt hij b.

Representatie van berichten

We legden de restrictie op dat het verzonden bericht een punt op de kromme moet zijn. Om de methode van ElGamal en Koblitz praktisch toe te kunnen passen, zullen we dus een goede manier moeten vinden om berichten te representeren als punten op de kromme. Alle punten aflopen en zo een getal toekennen zou te veel werk zijn, aangezien q normaal gesproken heel groot is. Dit is een lastig probleem en veel oplossingen zijn probabilistisch. Er is in dit geval een kleine kans dat bepaalde berichten niet verzonden kunnen worden omdat ze geen representatie als punt op de kromme hebben.

We zullen niet ingaan verder ingaan op dit representatieprobleem, want we zullen zien dat we het probleem ook op kunnen lossen door elliptische krommen op een andere wijze in de encryptie te betrekken.

6.2 ECIES

We kunnen de eerder genoemde aanpak combineren met een andere encryptiemethode om een hybride aanpak te krijgen. Als hierbij de juiste keuzes gemaakt worden, worden de voordelen van beide methoden gecombineerd. Dit biedt een oplossing voor het representatieprobleem en is aanzienlijk efficiënter voor lange berichten.

Een voorbeeld van een hybride methode is het “Elliptic Curve Integrated Encryption Scheme”, afgekort ECIES. Deze standaard wordt gespecificeerd in het document “SEC 1: Elliptic Curve Cryptography” (dit document is onder “Certicom” te vinden in de bronvermelding).

Voorbereiding

Zoals eerder moeten er een lichaam, een elliptische kromme en een punt op de kromme gekozen worden. Hiernaast moeten we nu ook een symmetrisch encryptiealgoritme, een sleutelafleidingsfunctie en een algoritme voor het bepalen van de message authentication code van tevoren kiezen. Hierbij zijn de sleutelafleidingsfunctie en message authentication code (MAC) cryptografische primitieven die we in de inleiding over cryptografie besproken hebben. De ECIES standaard geeft lijstjes van mogelijke keuzes. De gekozen parameters moeten algemeen bekend zijn en we gaan er dan ook van uit dat, naast Alice en Bob, ook Trudy ze kent.

Aanmaken van sleutels

Bob maakt de sleutels e en d aan op dezelfde manier als eerder en maakt e publiek bekend.

Encryptie

De encryptieprocedure is wat ingewikkelder dan in het eerdere geval. Alice moet de volgende stappen uitvoeren:

- Kies een tijdelijk sleutelpaar (d', e') op dezelfde manier als eerder. Hiervan is d' dus een element van $\mathbb{Z}$ en $e' := d'G$ een punt op de kromme. Wederom is d' een private key en e' een public key.
- Bereken $z \in GF(q)$ als de x-coördinaat van $d'e$. Dit is een vermenigvuldiging van een geheel getal (de tijdelijke private key) met een punt (Bob's public key) en moet dus gelezen worden als herhaalde optelling.
- Gebruik de sleutelafleidingsfunctie om twee sleutels, K_E en K_M, uit z af te leiden. Dit wordt gedaan door een langere sleutel K_{EM} aan te laten maken en deze te splitsen in twee stukken. Dit kan op deze wijze omdat K_{EM}, K_E en K_M allen simpele bitstrings zijn zonder verdere structuur.
- Gebruik het symmetrisch encryptiealgoritme om het bericht b met de sleutel K_E te versleutelen. We noemen het resultaat c.
- Bepaal de MAC voor het versleutelde bericht c met sleutel K_M.
- Verzend achtereenvolgens de publieke tijdelijke sleutel e', het versleutelde bericht c en de MAC.

Wat hier in feite gebeurt, is dat het bericht met een willekeurige sleutel symmetrisch wordt versleuteld. Om het toch te kunnen lezen wordt deze willekeurige sleutel met public key encryptie versleuteld en toegevoegd. Enkel Bob kan deze daardoor achterhalen. Het toevoegen van een MAC dient als bewijs dat het ontvangen bericht gelijk is aan het verstuurde bericht: iemand die de tijdelijke sleutel niet kan achterhalen kan namelijk geen nieuwe geldige MAC aanmaken.

We zien dat $\mathcal{P}$ en $\mathcal{C}$ gelijk zijn aan de verzamelingen van plaintexts en cyphertexts voor de symmetrische methode. Als we een methode kiezen die bitstrings omzet in bitstrings, is er geen probleem meer met de representatie van berichten als plaintexts. Alle gegevens worden door een computer namelijk beschouwd als bitstrings.

Decryptie

De decryptie door Bob gebeurt als volgt:

- Ontvang achtereenvolgens de publieke tijdelijke sleutel e', het versleutelde bericht c en de MAC.
- Bereken $z \in K$ als de x-coördinaat van de'. Merk hierbij op dat:

  $$de' = d(d'G) = (dd')G = (d'd)G = d'(dG) = d'e$$

  De gevonden z is dus gelijk aan de z die Alice eerder berekende.
- Gebruik de sleutelafleidingsfunctie om de twee sleutels, K_E en K_M, uit z af te leiden. Dit moet gebeuren op dezelfde wijze dat Alice dit deed.
- Controleer de MAC voor het versleutelde bericht c met de sleutel K_M. Als deze niet klopt, dan heeft iemand het bericht veranderd en wordt de decryptie met een foutmelding afgebroken.
- Gebruik het symmetrisch encryptiealgoritme om het bericht c met de sleutel K_E te ontcijferen. We vinden hiermee het bericht b weer terug.

Het idee achter het ontcijferen is dus dat we dankzij de structuur van de elliptische kromme met behulp van de private key de waarde van z kunnen achterhalen. Het is essentieel dat de publieke sleutel hiervoor alleen bruikbaar is, als ook de tijdelijke private key bekend is. Als noch de private key noch de tijdelijke private key bekend zijn, moet het heel moeilijk zijn om z en de daaruit afgeleide sleutels te bepalen. In het volgende hoofdstuk zullen we kijken waarom men denkt dat dit inderdaad heel moeilijk is.

6.3 Digitale handtekening

Voorbereiding

Ook voor het plaatsen van digitale handtekeningen kunnen we gebruik maken van elliptische krommen over eindige lichamen. Een voorbeeld van een methode hiervoor is ECDSA ("Elliptic Curve Digital Signature Algorithm"). Net als bij ECIES moeten we van tevoren een eindig lichaam $K = GF(p^n)$, een kromme $\mathcal{E}_a[K]$ en een punt $G \in \mathcal{E}_a[K]$ kiezen. Ook de gebruikte sleutelparen zijn hetzelfde als degene die voor encryptie gebruikt worden. Daarnaast moet nu ook een hash algoritme gekozen worden.

Plaatsen van handtekening

Een digitale handtekening is een paar $(r, s) \in (\mathbb{Z}/g\mathbb{Z})^2$ met $g := \#G$ de orde van G. De handtekening wordt als volgt berekend:

- Kies een tijdelijk sleutelpaar (d', e') op dezelfde manier als eerder. Hiervan is d' dus een geheel getal en $e' := d'G$ een punt op de kromme. Wederom is d' een private key en e' een public key.
- Kies voor r de x-coördinaat van e' modulo g. Als r de nulklasse is, dan moet een ander tijdelijk sleutelpaar gekozen worden. Als we deze waarde zouden toelaten, dan zou de handtekening niet meer van de private key afhangen en zou iedereen dus een geldige handtekening kunnen zetten.
- Gebruik de hash functie om een getal h af te leiden uit het te ondertekenen bericht b.
- Bereken $s := (d')^{-1}(h + r\,d)$ binnen de ring $\mathbb{Z}/g\mathbb{Z}$.

Het paar (r, s) kan nu samen met het bericht b bewaard worden.

Controle van handtekening

Als Bob de public key van Alice kent, dan kan hij de handtekening controleren. Merk op dat Bob niet, zoals bij een message authentication code (MAC), de handtekening zelf na kan maken. Dit kan alleen Alice, want daarvoor is haar private key nodig. De controle moet dus op een andere manier plaatsvinden.

We gaan ervan uit dat Bob het bericht p en de handtekening (r, s) heeft ontvangen. Verder kent hij de public key e van Alice. Hij volgt de volgende aanpak:

- Gebruik de hash functie om het getal h af te leiden uit het te ondertekenen bericht b.

- Bereken $u_1 := h s^{-1}$, $u_2 := r s^{-1}$ en tenslotte $R := u_1 G + u_2 e$.
- Schrijf r' voor de x-coördinaat van R.
- Accepteer de handtekening als $r' \equiv r \pmod{g}$ en weiger anders.

We kunnen narekenen dat dit werkt:

$$R = u_1 G + u_2 e = h s^{-1} G + r s^{-1} d G = s^{-1}(h + r d) G = d'(h + r d)^{-1}(h + r d) G = d' G = e'$$

We nemen aan beide kanten de x-coördinaat en we zien dat volgt $r' = r$.

7 Veiligheid

7.1 Wat is “heel moeilijk”?

We hebben eerder de vage term “heel moeilijk” een aantal malen gebruikt in verschillende contexten. Hier een overzicht:

1. (over encryptie) Zonder kennis van e moet het heel moeilijk zijn informatie over b af te leiden uit $E_e(p)$
2. (over public key encryptie) Zelfs met kennis van e moet het heel moeilijk zijn de inverse van E_e te berekenen
3. (over public key encryptie) Het moet ook heel moeilijk zijn om d uit e af te leiden
4. (over cryptografische hash) Het moet heel moeilijk zijn om uit het beeld van een bitstring onder een hash functie informatie over die bitstring af te leiden
5. (over ECIES) Als noch de private key noch de tijdelijke private key bekend zijn, moet het heel moeilijk zijn om z en de daaruit afgeleide sleutels te bepalen.

Op al deze plekken is deze formulering gebruikt omdat het woord “onmogelijk” hier niet haalbaar of niet wenselijk is. In de meeste van de hierboven genoemde gevallen kunnen we namelijk heel eenvoudig een algoritme vinden dat de taak die als “heel moeilijk” wordt omschreven, uit kan voeren.

Punt 1 is in dit rijtje een uitzonderingsgeval. We kunnen encryptiemethoden vinden waarvoor het bewijsbaar onmogelijk is om informatie over b af te leiden uit enkel $E_e(b)$. Een voorbeeld van zo'n methode is de reeds besproken one time pad. We hebben gezien dat aan deze methoden zodanige bezwaren kleven, dat ze praktisch vaak niet toepasbaar zijn. We namen daarom toch meestal genoegen met “heel moeilijk”.

Punt 3 levert een mooi voorbeeld waar we het “heel moeilijke” probleem gegarandeerd wel kunnen oplossen. Zij namelijk $e \in \mathcal{K}_e$ bekend en $d \in \mathcal{K}_d$, de bijbehorende decryptiesleutel, onbekend. $\mathcal{K}_d$ is aftelbaar, dus er bestaat een aftelling $(d_i)_{i \in \mathbb{N}}$. Loop nu $i \in \mathbb{N}$ door en controleer telkens of E_e en D_{d_i} elkaars inversen zijn. Als $d_i = d$, dan is dit het geval. Zo wordt d (of een andere sleutel d' zodanig dat $D_d = D_{d'}$) in eindige tijd gevonden. Deze aanpak wordt “brute force” genoemd.

7.2 De brute force aanval

We proberen het bepalen van d uit e door een brute force aanval moeilijk te maken, door te zorgen dat er veel rekentijd nodig is om de private key te vinden. Voor beide vormen van encryptie op basis van elliptische krommen is deze sleutel een geheel getal $d \in \mathbb{Z}$. Als we naar de algoritmen kijken, dan zien we dat dit getal enkel wordt gebruikt als vermenigvuldigingsfactor voor het punt G of een veelvoud daarvan. We krijgen zo enkel punten in de cyclische deelgroep $\{nG \mid n \in \mathbb{Z}\} \subseteq \mathcal{E}_a[K]$ voortgebracht door G. Het aantal sleutels dat echt onderling verschilt, is dus nooit groter dan het aantal elementen van deze groep, oftewel de orde van G. Om een brute force aanpak van het probleem moeilijk te maken, moeten we het lichaam, de kromme en G dus zodanig kiezen dat deze groep enorm groot is.

Een voor de hand liggende keuze is, om het lichaam K groter te maken door een hoge q te kiezen. Volgens de stelling van Hasse geldt $|\#\mathcal{E}_a[K]-(q+1)|\leq 2\sqrt{q}$ (Charlap en Robbins, 1988). Het aantal punten op de elliptische kromme neemt dus lineair toe in q. We moeten vervolgens in deze grote elliptische groep een element G zoeken met hoge orde.

De brute force aanval is meestal niet het grootste probleem. Door het onderliggende lichaam K voldoende groot te kiezen, kunnen we ervoor zorgen dat deze aanvallen ver buiten het bereik van de hedendaagse computers liggen. Als het lichaam q elementen heeft, dan kost het $\log_2 q$ bits om een element op te slaan. We noemen dit de sleutellengte. De rekentijd die een bewerking (zoals optelling of vermenigvuldiging) in het lichaam kost, hangt normaal gesproken polynomiaal af van deze sleutellengte. De tijd die een brute force aanval kost, hangt echter lineair van de orde van G af. De verwachting van het aantal te doorzoeken sleutels is namelijk de helft van het aantal mogelijke sleutels. Een brute force aanval kost dus een rekenkracht die exponentieel is in de sleutellengte. Keuze van een voldoende grote q is hierdoor voldoende om het brute force aanvallers moeilijk te maken.

In de praktijk is het overigens zelden nodig zelf parameters te genereren, aangezien er een aantal documenten zijn die geschikte waarden aangeven. Hiervan is “SEC2: Recommended Elliptic Curve Domain Parameters” een voorbeeld (dit document is onder “Certicom” te vinden in de bronvermelding). De gegeven waarden zijn zo goed mogelijk bestand tegen momenteel bekende aanvallen. Er worden parameters gegeven voor verschillende waarden van q, zodat men een balans kan vinden tussen veiligheid en snelheid.

7.3 Het discrete logaritme

Naast de brute force aanval is het denkbaar dat er algoritmen bestaan die de encryptie sneller kunnen breken. Er bestaat een aantal problemen waarvan vermoed wordt dat ze in het algemeen niet oplosbaar zijn op een manier die significant sneller is dan een brute force aanval. Door de veiligheid van het algoritme te herleiden tot zo'n probleem hebben we een zekere garantie dat we het Trudy voldoende moeilijk kunnen maken enkel door grote sleutels te gebruiken. Mocht het probleem echter alsnog opgelost worden, dan valt de basis achter deze veiligheid echter weg.

Een bekend voorbeeld van zo'n probleem, is het berekenen van het “discrete logaritme”: gegeven een eindige cyclische groep G met generator $b\in G$ en een element $g\in G$, zoek een $k\in\mathbb{Z}$ zodanig dat $g=k\,b$ (we gebruiken additieve notatie, dus dit is een herhaalde optelling). Er is geen efficiënte manier bekend om dit te doen.

Als we een oplossing vinden voor het discrete logaritme probleem, dan kunnen we onze encryptie eenvoudig breken. We kennen de public key $e=d\,G$ en om d te vinden kunnen dan dus simpelweg het discrete logaritme toepassen binnen de cyclische groep voortgebracht door G.

Om ook de implicatie de andere kant op de kunnen krijgen, moeten we een aangepaste vorm van het discrete logaritme probleem gebruiken. Deze staat bekend als het “Diffie-Hellman probleem” en kan als volgt geformuleerd worden: gegeven een eindige cyclische groep G met generator $b\in G$ en zekere $g_1, g_2\in G$, waarbij $g_1=d_1 b$ en $g_2=d_2 b$, zoek $g=(d_1 d_2)b$ (weer additieve notatie). Hierbij zijn $d_1, d_2\in\mathbb{Z}$ onbekend.

Als we het discrete logaritme kunnen berekenen, dan is een oplossing van Diffie-Hellman triviaal. Omgekeerd is het echter denkbaar dat we dit probleem op kunnen lossen zonder ook een oplossing te vinden voor het discrete logaritme. Equivalentie met Diffie-Hellman levert dus mogelijk een iets

zwakkere beveiliging dan equivalentie met het discrete logaritme probleem.

Als we de formulering van beide encryptiemethoden bekijken, zien we telkens punten van de vorm $d_1 b$ en $d_2 b$ openbaar zijn en dat $d_1 d_2 b$ met een private key achterhaald wordt. De veiligheid van onze encryptiemethoden is dus equivalent met de oplosbaarheid van het Diffie-Hellman probleem: als we de één op kunnen lossen, kunnen we die gebruiken om de ander ook op te lossen.

Deze equivalentie levert enige garantie voor de veiligheid ervan: als iemand de encryptie weet te breken, gaat het in ieder geval om een succesvol wiskundige. Het in dit geval te hopen dat deze wiskundige zijn (of haar) vinding publiceert in plaats van deze te misbruiken voor het breken van beveiligingen.

We zagen bij de ElGamal methode dat deze werkt voor algemene cyclische groepen. We hebben ook gezien dat rekenen met elliptische krommen relatief veel rekenwerk inhoudt. Het lijkt dus meer voor de hand te liggen om bijvoorbeeld cyclische deelgroepen van $\mathbb{Z}_p^*$ te gebruiken in plaats van elliptische krommen. De reden dat toch voor elliptische krommen wordt gekozen is gerelateerd aan het discrete logaritme probleem.

Momenteel is er een enigszins efficiënte aanpak van het discrete logaritme probleem bekend in het geval dat gerekend wordt binnen cyclische deelgroepen van $\mathbb{Z}_p^*$. Dit algoritme staat bekend als het "index calculus algorithm". Deze methode en haar efficiëntie worden beschreven in Odlyzko (1984). Terwijl de brute force methode exponentieel was in het aantal bits van b, is deze methode sub-exponentieel. Het ziet er echter naar uit dat de aanpak niet aan te passen is voor oplossen van het discrete logaritme probleem voor elliptische krommen. Silverman en Suzuki (1998) geven hier argumenten voor.

8 Vergelijking met andere methoden

8.1 Symmetrische encryptie

We zullen de behandelde symmetrische methoden gezamenlijk bespreken, want ze delen vergelijkbare eigenschappen. In het algemeen heeft symmetrische encryptie de volgende voordelen:

- Ze zijn meestal aanzienlijk sneller dan methoden voor public key encryptie.
- Ze laten meestal willekeurige bitstrings toe voor plaintexts en sleutels. Dit voorkomt representatieproblemen en maakt gebruik van een sleutelafleidingsfunctie mogelijk.

Een belangrijk nadeel is dat Trudy de encryptiesleutel niet mag kennen, aangezien deze gelijk is aan de decryptiesleutel.

In sommige situaties kunnen symmetrische encryptiealgoritmen zonder verdere aanpassing gebruikt worden om berichten veilig van Alice naar Bob te brengen. Aangezien $d=e$ mag de encryptiesleutel in dit geval niet aan derden bekendgemaakt worden.

Symmetrische methoden zijn bruikbaar als Alice en Bob van tevoren een sleutel afgesproken hebben over een kanaal dat niet afgeluisterd wordt. Verder zijn er methoden om een sleutel af te spreken over een gewoon kanaal zonder dat de sleutel uit de communicatie afgeleid kan worden. Een voorbeeld hiervan is het Diffie-Hellman sleuteluitwisselingsprotocol. Deze aanpak is gebaseerd op het discrete logaritme probleem.

In veel situaties zijn deze aanpakken niet geschikt. Gebruik van een sleuteluitwisselingsprotocol vereist interactie tussen verzender en ontvanger. Dit is traag en niet altijd mogelijk. Van tevoren afspreken van sleutels lukt nog wel als het alleen om Alice en Bob gaat. Als er echter ook berichten uitgewisseld moeten kunnen worden met Coby, Dirk, Ellen, Frank, Gerda en Hans, dan moet er voor elk paar personen een symmetrische sleutel afgesproken zijn. Als er n personen berichten uit willen wisselen, dan zijn er dus $\frac{1}{2}n(n-1)$ sleutels nodig. Voor deze acht personen zijn dat al 28 sleutels.

In gevallen waar symmetrische methoden niet geschikt zijn ligt het voor de hand een public key methode, zoals bijvoorbeeld het besproken ECIES, te gebruiken.

8.2 RSA

De eigenschappen van RSA komen sterk overeen met die van methoden voor encryptie met elliptische krommen. Om de twee te vergelijken, zullen we eerst naar de basis van de veiligheid van RSA kijken.

Veiligheid

De beveiliging van RSA is gebaseerd op het idee dat Trudy de waarde van $\phi(n)$ niet snel kan berekenen. Als zij deze waarde wel zou kunnen achterhalen, dan zou ze d eenvoudig uit e kunnen

berekenen. Trudy zou $\phi(n)$ kunnen bepalen door n te ontbinden in priemfactoren. Er is echter geen manier bekend om dit snel te doen, net zoals dat voor het Diffie-Hellman probleem op elliptische krommen het geval was. De veiligheid van RSA rust op de onmogelijkheid om grote getallen snel te factoriseren. Het is echter denkbaar dat er nog andere aanvallen mogelijk zijn waarbij n niet gefactoriseerd wordt. Er is dus, in tegenstelling tot ECIES, geen equivalentie.

Vergelijking

De RSA methode is eenvoudiger en sneller dan encryptie met elliptische krommen. Vermenigvuldigingen binnen $\mathbb{Z}/n\mathbb{Z}$ kosten namelijk minder bewerkingen dan optellingen op elliptische krommen.

Hybride encryptie, zoals bij ECIES, is ook mogelijk voor RSA. Het idee is hetzelfde: er wordt een willekeurige sleutel gebruikt voor symmetrische encryptie van het bericht en deze sleutel wordt met public key encryptie beschermd. Aangezien we meestal een sleutel van vaste lengte gebruiken, zoals bij EAS, is voor lange berichten de rekentijd die nodig is voor de public key encryptie verwaarloosbaar ten opzichte van de rekentijd voor symmetrische encryptie van het bericht. Dit betekent dat het in de praktijk weinig uitmaakt dat elliptische kromme cryptografie iets langzamer is dan RSA.

Beide methoden bieden naast encryptie ook een manier om digitale handtekeningen te zetten.

Het belangrijkste verschil zit hem in de basis van de veiligheid van beide methoden:

- Hoewel zowel het Diffie-Hellman probleem als het factoriseringsprobleem nog niet opgelost zijn, worden snellere oplossingen voor het factoriseren van getallen waarschijnlijker geacht dan snelle discrete logaritmen op de groep van elliptische krommen.
- Breken van encryptie met elliptische krommen is equivalent met het oplossen van het Diffie-Hellman probleem. Oplossen van het factoriseringsprobleem impliceert het breken van RSA, maar de tegengestelde implicatie hoeft hier niet waar te zijn.

Dit verschil leidt er toe dat de aangeraden minimale sleutelgrootte voor RSA hoger ligt dan voor elliptische kromme cryptografie. Momenteel biedt volgens RSA Security een elliptische methode met een sleutelgrootte van 160 bits vergelijkbare bescherming als RSA met een 1024-bit sleutel.

RSA biedt momenteel voldoende veiligheid en heeft in veel gevallen de voorkeur. Op de lange termijn is het echter te verwachten dat elliptische kromme cryptografie langer veilig blijft dan RSA. Om deze reden raadt de NSA (een Amerikaanse veiligheidsdienst die zich primair bezighoudt met cryptografie) bedrijven aan over de schakelen van RSA naar cryptografie gebaseerd op elliptische krommen.

9 Conclusie

We hebben gezien dat elliptische krommen goed bruikbaar zijn voor het beveiligen van gegevens. Methoden gebaseerd op elliptische krommen kunnen gebruikt worden om alle hoofddoelen van de cryptografie te kunnen bereiken: authenticatie, integriteit en confidentialiteit.

Wanneer we encryptie gebaseerd op elliptische krommen combineren met symmetrische encryptie, krijgen we een sterk encryptiealgoritme. De veiligheid hiervan is equivalent met onoplosbaarheid van het Diffie-Hellman probleem. Zolang dit probleem niet opgelost wordt, zal een tegenstander over een rekenkracht moeten beschikken die exponentieel is in de sleutellengte.

De kracht van encryptie met elliptische krommen wordt over het algemeen als sterker ingeschat dan die van andere public key systemen, zoals het veel gebruikte RSA. Hier staat tegenover dat elliptische krommen wel lastiger in het gebruik zijn en meer rekentijd kosten. Bij gebruik van hybride encryptie op lange berichten is het verschil in snelheid echter te verwaarlozen.

Symmetrische algoritmen bieden eigenschappen die vaak prettiger zijn dan die van encryptie met elliptische krommen. Tot de mogelijkheden behoort zelfs perfecte encryptie. Gebruik van deze algoritmen is in veel gevallen echter niet mogelijk of niet handig omdat sleuteldistributie aanzienlijk lastiger is dan bij public key encryptie.

Al met al zien we dat encryptie met behulp van elliptische krommen vooral nuttig is in situaties waar aan alle onderstaande voorwaarden voldaan is:

- Er is duurzame veiligheid nodig;
- Er is voldoende rekenkracht of tijd beschikbaar;
- Symmetrische encryptiemethoden zijn niet geschikt.

10 Bronvermelding

- Certicom Research, *Standards for Efficient Cryptography (SEC) 1: Elliptic Curve Cryptography*, 2000, http://www.secg.org/download/aid-385/sec1_final.pdf
- Certicom Research, *Standards for Efficient Cryptography (SEC) 2: Recommended Elliptic Curve Domain Parameters*, 2000, http://www.secg.org/collateral/sec2_final.pdf
- L.S. Charlap en D.P. Robbins, *An Elementary Introduction to Elliptic Curves*, CRD Expository Report no. 31, december 1988. Ook beschikbaar op internet: http://www.idaccr.org/reports/er31.ps
- T. Elgamal, *A public key cryptosystem and a signature scheme based on discrete logarithms*, IEEE Transactions on Information Theory, juli 1985, volume: 31, issue: 4, pp. 469-472
- B. S. Kaliski Jr., *Elliptic Curves and Cryptography: A pseudorandom Bit Generator and Other Tools*, 1988, http://www.rsasecurity.com/rsalabs/node.asp?id=2018
- N. Koblitz, *Elliptic Curve Cryptosystems*, Mathematics of Computation, volume 48, januari 1987, no. 177, pp. 203-209
- A. J. Menezes, P. C. van Oorschot en S. A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1996. Ook beschikbaar op internet: http://www.cacr.math.uwaterloo.ca/hac/
- NSA, *The Case for Elliptic Curve Cryptography*, opgevraagd 19 juni 2006, http://www.nsa.gov/ia/industry/crypto_elliptic_curve.cfm
- A. M. Odlyzko, *Discrete logarithms in finite fields and their cryptographic significance*, Advances in Cryptology: Proceedings of EUROCRYPT '84, 1984, pp. 224-314. Ook beschikbaar op internet: http://www.dtc.umn.edu/~odlyzko/doc/arch/discrete.logs.pdf
- RSA Security, How do elliptic curve cryptosystems compare with other cryptosystems, Crypto FAQ, opgevraagd 2 juli 2006, http://www.rsasecurity.com/rsalabs/node.asp?id=2245
- J. Silverman en J. Suzuki, *Elliptic curve discrete logarithms and index calculus*, Advances in Cryptology, ASIACRYPT '98, Lecture Notes in Computer Science 1514, 1998, Springer, pp. 110-125.
- P. Stevenhagen, *Algebra III*, 2006, http://websites.math.leidenuniv.nl/algebra/algebra3.pdf
- A. S. Tanenbaum, *Computer Networks*, Prentice Hall, 2003.
- E. W. Weisstein, *Elliptic Curve*, MathWorld, a Wolfram Web Resource, 16 april 2003, http://mathworld.wolfram.com/EllipticCurve.html
- Wikipedia contributors, *Advanced Encryption Standard*, Wikipedia, The Free Encyclopedia, 29 juni 2006; http://en.wikipedia.org/w/index.php?title=Advanced_Encryption_Standard&oldid=61261787
- Wikipedia contributors, *Cryptographic hash function*, Wikipedia, The Free Encyclopedia, 10 juli 2006; http://en.wikipedia.org/w/index.php?title=Cryptographic_hash_function&oldid=63053373
- Wikipedia contributors, *Elliptic curve*, Wikipedia, The Free Encyclopedia, 20 mei 2006; http://en.wikipedia.org/w/index.php?title=Elliptic_curve&oldid=54217485

- Wikipedia contributors, *Elliptic curve cryptography*, Wikipedia, The Free Encyclopedia, 12 juni 2006; http://en.wikipedia.org/w/index.php?title=Elliptic_curve_cryptography&oldid=58238196
- Wikipedia contributors, *Euler's totient function*, Wikipedia, The Free Encyclopedia, 6 juni 2006; http://en.wikipedia.org/w/index.php?title=Euler%27s_totient_function&oldid=57212029
- Wikipedia contributors, *Finite field arithmetic*, Wikipedia, The Free Encyclopedia, 20 juni 2006; http://en.wikipedia.org/w/index.php?title=Finite_field_arithmetic&oldid=60831745
- Wikipedia contributors, *Hill cipher*, Wikipedia, The Free Encyclopedia, 12 juni 2006; http://en.wikipedia.org/w/index.php?title=Hill_cipher&oldid=52305156
- Wikipedia contributors, *Weierstrass's elliptic functions*, Wikipedia, The Free Encyclopedia, 16 juni 2006; http://en.wikipedia.org/w/index.php?title=Weierstrass%27s_elliptic_functions&oldid=58966387