Digitale geldtransacties

Stefanie Romme
3013170
Wiskunde, Bachelor
Begeleider: Wieb Bosma

Radboud Universiteit Nijmegen
5 juli 2012

Samenvatting

Sinds de opkomst van het internet zijn elektronische geldtransacties onmisbaar geworden. Daarom moet het internetbankieren goed beveiligd worden met zowel symmetrische als asymmetrische cryptografie. Zo wordt er gezorgd voor een goede toegangscontrole, authenticatie en verantwoording om onbevoegden niet de mogelijkheid te geven om binnen te dringen. Daarnaast spelen integriteit en privacy een belangijke rol in de communicatie. Hiervoor wordt gezorgd door middel van hashfuncties, 3DES en RSA.

Inhoudsopgave

Hoofdstuk 1

Inleiding

Vroeger kende men nog geen geld. Door middel van directe ruilhandel werd aan de behoeften voldaan. Omdat de bakker niet altijd naar de kapper ging als de kapper brood nodig had, ging men rond 2500 voor Christus gebruik maken van waardevolle objecten, zoals metalen. Hierbij had men steeds een weegschaal nodig om de juiste hoeveelheid te bepalen, waardoor men op het idee kwam om munten te maken met een precies afgemeten hoeveelheid materiaal. Door de toegenomen welvaart gingen mensen aan het eind van de Middeleeuwen over grotere hoeveelheden munten beschikken. Vanwege het gesleep en tegen diefstal van deze munten werden banken opgericht waar men geld in bewaring kon geven tegen een goudbon waarop de afgegeven hoeveelheid geld stond vermeld. Later werden deze goudbonnen overdraagbaar, waardoor de bankbiljetten zijn ontstaan. De grote verandering dat er direct via de bank kon worden betaald vond plaats in de jaren '60, waarna ook de creditcard werd uitgevonden.
Na de opkomst van het internet eind jaren '90, werd het ook mogelijk om digitale geldtransacties uit te voeren. Deze transacties moeten uiteraard veilig verlopen. De wiskunde, en dan met name de cryptografie, speelt hier een belangrijke rol in. In deze scriptie zoeken we naar de wiskunde achter het internetbankieren. Allereerst wordt het begrip cryptografie besproken, waarna er bij de drie bekendste banken van Nederland wordt gekeken welke cryptografie zij toepassen om de geldtransacties veilig te laten verlopen. Vervolgens wordt er dieper ingegaan op de belangrijkste wiskundige methodes die bij het internetbankieren worden gebruikt, namelijk hashfuncties, DES/3DES en RSA.
Ik heb voor dit onderwerp gekozen, omdat ik graag een praktische toepassing van wiskunde wilde onderzoeken, waar iedereen mee te maken heeft. Cryptografie vind ik erg interessant, waardoor ik nieuwsgierig was naar de wiskundige achtergrond van het internetbankieren.

Hoofdstuk 2

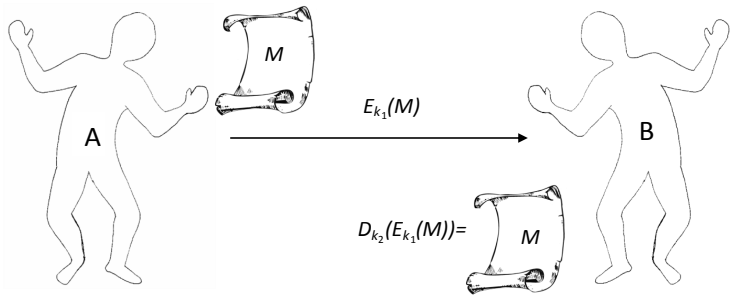
Cryptografie

2.1 Wat is cryptografie?

De cryptografie is een onderdeel van de cryptologie. Het woord cryptologie betekent de leer van het verbergen. Cryptografie houdt zich bezig met technieken om data te vercijferen, waarbij veelal gebruik wordt gemaakt van geheime sleutels. Alleen diegene die over de geheime sleutels beschikt, is in staat de vercijferde informatie te ontcijferen. Voor ieder ander is dit ondoenlijk. Hierdoor is het mogelijk om informatie alleen leesbaar te maken voor diegenen die het mogen lezen en tevens om de echtheid van bepaalde informatie aan te tonen. Met andere woorden, cryptografie is er voor privécommunicatie. Grote hoeveelheden persoonlijke en gevoelige informatie zijn tegenwoordig bevat in geautomatiseerde databanken en telefoonlijnen. Mede dankzij deze ontwikkelingen is cryptografie steeds belangrijker geworden.

Voor communicatie heb je een zender A (Alice) nodig die een bericht M wil sturen naar een ontvanger B (Bob). Dit bericht bestaat uit klare tekst, dat betekent dat het een stel woorden bevat uit een eindig alfabet $\{a, b, c, \ldots, z\}$. Om er voor te zorgen dat alleen Bob dit bericht kan lezen, wordt het bericht versleuteld. Dit versleutelen noemt men ook wel encrypten (E). Voor deze encryptie wordt een sleutel k_1 gebruikt die de zender kent. Na de encryptie stuurt A het geëncrypte bericht, ookwel de cijfertekst genoemd, naar B. Vervolgens zal B het moeten ontcijferen om het bericht M te kunnen lezen. Dit ontcijferen noemt men decrypten (D). Ook voor deze decryptie wordt gebruik gemaakt van een sleutel k_2.

Schematisch weergegeven:



Definitie 2.1. *Een cryptosysteem is een vijf-tupel* $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ *met:*

$\mathcal{P}$ *De klare tekst ruimte (plaintext)*
$\mathcal{C}$ *De cijfertekst ruimte (ciphertext)*
$\mathcal{K}$ *De sleutel-ruimte (key)*
$\mathcal{E}$ *Het encryptie algoritme*
$\mathcal{D}$ *Het decryptie algoritme*

waarbij $E_k : \mathcal{P} \to \mathcal{C},\ M \mapsto E(M)$ *de functie is die een bericht codeert en* $D_k : \mathcal{C} \to \mathcal{P},\ E(M) \mapsto M$ *de functie die een versleuteld bericht decodeert.*

Enkele eisen waar deze functies aan moeten voldoen zijn:

- Decryptie op een geëncrypt bericht M geeft M terug. Met andere woorden, voor alle $M \in \mathcal{P}$ geldt: $D_{k_1}(E_{k_2}(M)) = M$.
- $E_{k_1}(M)$ en $D_{k_2}(Y)$ zijn makkelijk te berekenen als k_1 en k_2 bekend zijn.
- Bij een publieke E_{k_1} is het niet makkelijk om k_2 en dus D_{k_2} te berekenen.

De eerste eis is logisch, want wanneer een decryptie op een geëncrypt bericht niet M oplevert, is het bericht nog niet ontcijferd. Dit betekent dat D een onjuist decryptie algoritme is.
De tweede eis is van belang om snel te communiceren. Als het encryptie of decryptie algoritme bekend is, moet met een gegeven bericht, respectievelijk cijfertekst, gemakkelijk de encryptie of decryptie gedaan kunnen worden.
De laatste eis is gebasseerd op de veiligheid, zodat alleen de ontvanger het bericht makkelijk kan lezen. Iemand die de cijfertekst heeft weten te bemachtigen, moet niet makkelijk in staat zijn om het bericht te decrypten. Hiervoor is het van belang dat het niet makkelijk is om het decryptie algoritme te kunnen berekenen, als het encryptie algoritme bekend is.

Een functie die aan bovenstaande eisen voldoet is een *trap-door one-way* functie. Hierin betekenen *trap-door* en *one-way* het volgende:

- *Trap-door*; de inverse functie is gemakkelijk te berekenen als de privé trap-door informatie (de sleutel) bekend is.
- *One-way*; $f : X \to Y$ heet een éénrichtingsfunctie als $f(x)$ gemakkelijk te berekenen is voor gegeven $x \in X$, maar voor $y \in Y$ $(= f(x))$ is het lastig om x te bepalen. Met andere woorden, het is makkelijk om één richting te berekenen, maar erg moeilijk om de andere richting te berekenen.

Definitie 2.2. *Een* bit *is de kleinste eenheid van informatie voor de computer, een 0 of 1.*

Definitie 2.3. *Een* byte *is een groep van acht bits die samen één teken vormen in het geheugen van een computer.*

In het vervolg van dit hoofdstuk wordt er onderscheid gemaakt tussen symmetrische en asymmetrische cryptografie. Vervolgens bekijken we de beveiligingsaspecten die een rol spelen bij de keuze voor het beveiligen.

2.2 Symmetrische cryptografie

Bij symmetrische algoritmen bestaat de encryptie uit een reeks omkeerbare stappen. Daarom is hierbij de sleutel voor vercijfering hetzelfde als voor ontcijfering. In wiskundige notatie:

$$E_k(\mathcal{P}) = \mathcal{C}$$

$$D_k(\mathcal{C}) = \mathcal{P}$$

En dus geldt ook:

$$D_k(E_k(\mathcal{P})) = \mathcal{P}$$

Het is gebaseerd op het idee dat beide partijen een geheim delen, het geheim hoe de data gecodeerd is en hoe de data dus ook weer te decoderen valt.

Met symmetrische cryptografie kan de authenticatie worden nagegaan. *Authenticatie* is het bewijzen van je identiteit aan een ander. Omdat bij symmetrische cryptografie niemand anders dan Bob en Alice de sleutel k kent, en Bob van zichzelf weet dat hij de boodschap niet zelf heeft verstuurd, kan Bob er op vertrouwen dat de boodschap van Alice afkomstig is.

Een nadeel van symmetrische systemen is dat *authenticatie* verdwijnt bij het communiceren met meerdere mensen. Dit komt doordat de andere partijen dezelfde verzendsleutel hebben, waardoor niet meer is na te gaan wie de

zender van het bericht is. Voor dergelijke authenticatie gebruikt men vaak asymmetrische cryptografie. Om de *authenticatie* te behouden met symmetrische systemen zijn er y sleutels nodig voor ieder groepje van n personen om allemaal individueel met elkaar te kunnen communiceren, met $y = n(n-1)/2$. De hoeveelheid sleutels zal dan enorm snel oplopen.
De symmetrische cryptografie wordt toegepast in: 3DES (Triple Data Encryption Standard), AES (Advanced Encryption Standard), IDEA (International Data Encryption Algorithm) en RC4 (Rivest Cipher 4). Bij digitale geldtransacties worden 3DES en RC4 gebruikt. Deze zullen dan ook uitgebreider besproken worden in Hoofdstuk 3 en 4.

2.3 Asymmetrische cryptografie

In asymmetrische algoritmen gebruikt men niet één, maar twee sleutels. De ene is openbaar en de andere is geheim (alleen de eigenaar van de sleutel kent deze). In het Engels heten deze sleutels public key en private key, waardoor asymmetrische cryptografie ook wel public-key cryptografie wordt genoemd. Wanneer A een bericht wil sturen naar B hoeft A het bericht alleen te vercijferen met de openbare sleutel van B en kan alleen B het decrypten. Doordat de encryptie sleutel openbaar is, kan iedereen een bericht versleutelen. Decryptie kan alleen gedaan worden door de eigenaar van de openbare sleutel. Ook andersom is dit mogelijk, waarbij de geheime sleutel wordt gebruikt om te coderen en met de publieke sleutel kan worden gedecodeerd. Dit wordt gebruikt bij het digitaal ondertekenen van berichten. In wiskundige notatie:

$$E_{k_1}(\mathcal{P}) = \mathcal{C}$$

$$D_{k_2}(\mathcal{C}) = \mathcal{P}$$

En dus geldt ook:

$$D_{k_2}(E_{k_1}(\mathcal{P})) = \mathcal{P}$$

Belangrijk is hierbij dat uit de encryptiesleutel niet is te zien hoe de berekening van de decryptie verloopt of omgekeerd. Hierdoor blijft één sleutel geheim.

Voordelen van asymmetrische cryptografie zijn:

- Men kan zelf kiezen wie de versleutelde informatie kan lezen (bijvoorbeeld bij een digitale handtekening) of wie allemaal informatie kan versleutelen, omdat men zelf de openbare sleutel verstrekt.

- Er kan een digitale handtekening worden gezet onder een bericht, waarmee wordt bewezen dat een bericht ongewijzigd afkomstig is van degene die het ondertekend heeft (*integriteit* en *authenticatie*). Het plaatsen van een digitale handtekening vraagt kennis van een geheime encryptie sleutel, zodat deze handtekening maar door één persoon kan worden gezet. Om de handtekening te controleren is het voldoende om over de bijbehorende openbare decryptie sleutel te beschikken.

2.4 Beveiligingsaspecten

Onderstaande paragraaf komt overeen met het boek *Basismethoden Cryptografie* (J.C.A. van der Lubbe, 1994).

Cryptografie wordt toegepast om informatie te beveiligen. Deze informatie kan uiteenlopen van een simpel berichtje, een wachtwoord, tot bank gegevens of zelfs tv kanalen. De tv exploitant wil namelijk zoveel mogelijk illegaal kijken voorkomen. Deze toepassingen kunnen worden verdeeld in twee categorieën, namelijk opslag en transport van informatie. Het beveiligingsniveau moet bij de opslag van informatie hoger liggen, omdat dit vaak voor lange duur is en deze informatie ook langere tijd waarde zal hebben. Er is dus meer tijd voor een aanval. Het beveiligingsniveau hangt ook af van de investering die gedaan wordt om de gewenste beveiligingsmaatregelen te nemen. Vanuit het economisch oogpunt mogen de kosten niet te hoog liggen, zeker wanneer de beveiliging alleen voor het bedrijf van belang is en niet voor de consument. Daarnaast zijn er nog een aantal invloedrijke factoren van belang bij de keuze voor beveiliging, namelijk het doel, de soort en de dimensie.

2.4.1 Het doel

Als men wil beveiligen tegen het uitlezen en afluisteren van data is het belangrijk dat alleen diegene die gemachtigd is deze data te benaderen, de data ook daadwerkelijk kan benaderen. Met andere woorden, *vertrouwelijkheid* is een belangrijke eis. Dit hangt nauw samen met geheimhouding van het sleutelbeheer.
Wil men beveiligingsmaatregelen tegen het manipuleren en aantasten van data en ook tegen het ongeoorloofd gebruik van netwerk, dan gaat het om *betrouwbaarheid*. Aan deze eis kan worden voldaan door gebruik te maken van integriteit en authenticatie.

2.4.2 De soort

Er zijn verschillende soorten beveiliging, namelijk *fysieke beveiliging*, *hardware- en softwarematige beveiliging* en *organisatorische beveiliging*.
Bij fysieke beveiliging worden er maatregelen opgesteld om het fysiek binnendringen in systemen tegen te gaan. Dit kan bijvoorbeeld worden gedaan door temperatuurs- en trillingssensoren. Bij hardware- en softwarematige beveiliging wordt gebruik gemaakt van cryptografische algoritmen en methoden. Dit is waar we gebruik van maken bij elektronische geldtransacties. Door organisatorische beveiliging worden er randvoorwaarden gecreëerd waardoor de genomen fysieke en hardware- en softwarematige beveiligingsmaatregelen effectief kunnen zijn.

2.4.3 De dimensie

Bij het toepassen van cryptografie gaat men vooral *preventief* te werk. Dat wil zeggen dat men sterke cryptografische algoritmen en procollen gebruikt en tevens goede fysieke en organisatorische maatregelen neemt om de kans dat er iets gebeurt te minimaliseren. Toch is absolute beveiliging onmogelijk. Daarom is het *schade-beperkende* aspect ook erg belangrijk. De schade die toch nog wordt geleden, moet zo snel mogelijk worden hersteld. Dit komt kijken bij het laatste aspect, het *corrigerende* aspect.

Hoofdstuk 3

Internetbankieren

Als we kijken naar de beveiligingsaspecten bij internetbankieren, kunnen we deze in onderstaande categorieën onderscheiden:

1. *Toegangscontrole*; is de persoon gerechtigd om de actie te ondernemen;
2. *Authenticatie*; is de persoon wel wie hij zegt dat hij is;
3. *Integriteit*; is ontvangen informatie wel dezelfde als verzonden;
4. *Privacy*; is de informatie onzichtbaar voor anderen;
5. *Verantwoording*; is het aantoonbaar dat een transactie daadwerkelijk heeft plaatsgevonden.

We kijken nu of aan deze beveiligingsaspecten wordt voldaan, door te kijken wat er cryptografisch precies gebeurt bij het internetbankieren. Allereerst kijken we wat er gebeurt als je inlogt en vervolgens kijken we wat er gebeurt als je geld overmaakt. Dit verschilt per bank, daarom kijken we alleen naar de drie bekendste banken van Nederland: de Rabobank, de ING bank en de ABN AMRO bank.

3.1 Inloggen

Om je bankrekening te bekijken en vervolgens geld over te maken, moet je eerst inloggen op de site van je bank. Hieronder staat per bank eerst beschreven welke handelingen je zelf moet verrichten en daarna staat beschreven welke cryptografische stappen er in de tussentijd daadwerkelijk zijn uitgevoerd en of er aan de eerste vier beveiligingsaspecten voldaan wordt. Aan het laatste aspect *verantwoording* hoeft de bank niet te voldoen, omdat er bij het inloggen geen geldtransactie plaatsvindt.

3.1.1 Rabobank

De achtereenvolgende handelingen die je bij de Rabobank zelf moet uitvoeren om in te loggen zijn:

1. Rekeningnummer intypen;
2. Pasnummer intypen;
3. Random reader:
   (a) Bankpas invoeren;
   (b) Druk op I (van inloggen);
   (c) Pincode invoeren;
   (d) Toets OK;
4. Transactiecode die op de random reader verschijnt intypen op de site;
5. Klik op Inloggen

Het eerste wat gebeurt tijdens het inloggen is het controleren van de pincode, met behulp van de niet-persoonsgebonden random reader. Dit is een offline proces waarbij de pincode onversleuteld naar de chip van de bankpas wordt gestuurd. De chip weet de pincode en kan deze dus verifiëren. Klopt de pincode niet, dan wordt er een teller met één verhoogd en geeft de chip een foutmelding. Wanneer er een juiste pincode wordt ingevoerd, gaat deze teller weer naar nul. Nadat er drie keer achtereen een verkeerde pincode wordt ingevoerd, accepteert de chip geen enkele pincode meer, waardoor de kaart is geblokkeerd. Hierdoor kan er niet meer ingelogd worden, ook niet met een andere random reader. Door deze offline controle wordt de *toegangscontrole* vastgesteld. Het bezitten van de pinpas en de random reader en het kennen van de pincode wordt gezien als het feit dat iemand gerechtigd is om de rekening te bekijken.

Vervolgens 'maakt' de random reader een transactiecode die slechts één keer te gebruiken is. Deze ontstaat door het getal nul samen met een teller, die in de chip van pinpas zit en het aantal transacties telt, te hashen en vervolgens met 3DES te encrypten (zie hoofdstuk 4 voor hashfuncties en 3DES). De samenstelling van het getal nul met de teller is simpelweg het achter elkaar plakken van deze getallen. We weten dat deze samenstelling niet tijds- of rekening-afhankelijk is. De berekende 3DES-waarde is iedere keer uniek, omdat de teller bij iedere transactie uniek is. De hashfunctie zorgt er voor de *authenticatie* en *integriteit*. De transactiecode die uiteindelijk verschijnt is

een samenstelling van bepaalde bits van dit getal, $3DES_k(\text{hash}(0Y))$, met Y de teller. Deze samenstelling is per bank hetzelfde.

Voordat de bank de transactiecode kan controleren, moet deze veilig worden gecommuniceerd met de bank. Dit gebeurt met *TLS* (Transport Layer Security), de opvolger van *SSL* (Secure Sockets Layer). Het enige verschil is dat TLS sterkere encryptie algoritmen gebuikt. TLS is een protocol dat zorgt voor een veilige verbinding met de server, waarbij *authenticatie*, *integriteit* en *privacy* worden gegarandeerd. In hoofdstuk 4 staat beschreven hoe dit protocol precies werkt.

Tot slot controleert de bank de ontvangen transactie. Omdat iedere pas een unieke k heeft, die is gebruikt om de 3DES waarde te berekenen, moet de bank eerst deze k bepalen. Dit gebeurt doordat de bank in het bezit is van een *masterkey*. Uit deze streng geheime masterkey kan de bank met behulp van je rekeningnummer en pasnummer, je persoonlijke sleutel k afleiden. Hiermee kan de bank de transactiecode ook berekenen en vervolgens vergelijken met de ontvangen waarde. Door het verifiëren van deze code weet de bank zeker dat de eigenaar van de pas inlogt, omdat deze code afhankelijk is van het rekeningnummer, het pasnummer en de persoonlijke sleutel k.

3.1.2 ING bank

De achtereenvolgende handelingen die je bij de ING bank zelf moet uitvoeren om in te loggen zijn:

1. Gebruikersnaam intypen;
2. Wachtwoord intypen;
3. Klik op Inloggen

Deze handelingen zijn persoonsgebonden. Als je een gebruikersnaam en het bijbehoorde wachtwoord kent, ben je gerechtigd om in te loggen (*toegangscontrole*). Nadat er op Inloggen is geklikt worden deze gegevens verstuurd naar de bank. Dit gebeurt opnieuw met een TLS verbinding, waarmee *authenticatie* en *integriteit* wordt aangetoond en de *privacy* behouden blijft. Vervolgens zal de bank deze ingevoerde gegevens controleren. Hoe is dit precies gebeurt is niet bekend, maar waarschijnlijk heeft de bank een hash van het wachtwoord opgeslagen waarmee ze het ingevoerde wachtwoord kunnen verifiëren.

Voor extra veiligheid bestaat de gebruikersnaam uit minstens acht tekens. Je ontvangt de gebruikersnaam in eerste instantie van de bank, maar kun je daarna zelf veranderen. Daarnaast moet het wachtwoord minstens één hoofdletter, één leesteken en één cijfer bevatten, waardoor een wachtwoord niet gemakkelijk te raden is.

3.1.3 ABN AMRO bank

De achtereenvolgende handelingen die je bij de ABN AMRO bank zelf moet uitvoeren om in te loggen zijn:

1. Klik aan welke e.dentifier je hebt;
2. Rekeningnummer intypen;
3. Pasnummer intypen;
4. Druk op OK;
5. E.dentifier:
   (a) Bankpas invoeren;
   (b) Pincode invoeren;
   (c) Toets OK;
   (d) Typ de code van de site in op je e.dentifier (afhankelijk van de e.dentifier);
6. Response die op de e.dentifier verschijnt intypen op de site;
7. Klik op OK

Deze handelingen komen bijna geheel overeen met de handelingen die bij de Rabobank worden uitgevoerd. Alleen spreken we hier van een e.dentifier in plaats van een random reader en wordt er hierbij al een code op de site weergegeven die je in moet voeren op de e.dentifier. Ook de e.dentifier is niet-persoonsgebonden. De onderliggende cryptografie is daarom vrijwel geheel hetzelfde.

De controle van de pincode vindt opnieuw onversleuteld plaats, waarmee de *toegangscontrole* wordt uitgevoerd. Vervolgens wordt er een transactiecode/response gemaakt, maar in dit geval gebeurt dit niet aan de hand van het getal nul. De code die wordt afgebeeld op de website en vervolgens wordt ingetypt op de e.dentifier wordt hier bij gebruikt. Het zichtbare random getal is

gegeneerd door de website van de bank. De web applicatie van de bank houdt bij wanneer dit random getal is gegenereerd. Hierdoor is dit getal dus tijd en transactie gebonden, dit zorgt voor een extra beveiliging. Vervolgens wordt dit getal N samen met de teller Y, de input van een hashfunctie, zodat de *authenticatie* en *integriteit* kan worden aangetoond. Daarna wordt de hash-waarde versleuteld met 3DES. Hierdoor ontstaat opnieuw een transactiecode $3\text{DES}_k(\text{hash}(NY))$. Deze wordt net zo gecommuniceerd en gecontroleerd als de Rabobank dat doet, waardoor opnieuw de *privacy*, *authenticatie* en *integriteit* verzekerd wordt.

3.2 Geld overmaken

Bij iedere bank moet je de volgende handelingen verrichten. Je typt het bedrag in dat je over wilt maken en geeft aan van welk rekeningnummer het geld af moet en naar welke het toe moet. Tevens vul je de naam van de ontvanger en eventueel een omschrijving in. Klik aan dat de transactie moet gebeuren, controleer de ingevulde gegevens en bevestig deze. Vervolgens verschilt het weer per bank wat er gebeurt, waarbij nu de *verantwoording* wel van belang is.

3.2.1 Rabobank

De achtereenvolgende handelingen die je bij de Rabobank zelf moet uitvoeren om geld over te maken zijn:

1. Random reader;
    (a) Bankpas invoeren;
    (b) Druk op S (van signeren);
    (c) Pincode intypen;
    (d) Typ de code van de site in op de random reader;
    (e) Toets OK;
    (f) Nogmaals OK (Bij bedragen boven de 750 euro moet je nog een tweede code invoeren);
2. Typ de transactiecode die op de random reader verschijnt in op de site;
3. Klik op Verzenden

Bij geldtransacties van de Rabobank gebeurt bijna hetzelfde als bij het inloggen bij de ABN AMRO bank, er zijn slechts twee verschillen. Het eerste verschil is dat je nu niet meer je rekeningnummer en je pasnummer in hoeft te voeren. Aangezien je al bent ingelogd, weet de bank de persoonlijke sleutel k en kan deze de transactiecode opnieuw controleren, omdat de bank de tijd en transactie gebonden code N ook weet. Het tweede verschil is dat je bij bedragen boven de 750 euro een tweede code M over moet nemen van het scherm. Ook deze code is tijd en transactie gebonden, waardoor dit een extra beveiliging geeft. Deze tweede code wordt achter de eerste code en de teller geplakt. Na deze samenstelling wordt de transactiecode op dezelfde manier berekend als bij het inloggen, dus door eerst te hashen en vervolgens de 3DES-waarde te berekenen. Hierdoor wordt net zo aan de eerste vier beveiligingsaspecten voldaan, als het inloggen bij de ABN AMRO bank.

Omdat we nu wel te maken hebben met een geldtransactie moet er ook aan het laatste beveiligingsaspect, *verantwoording*, worden voldaan. Voor de bank wordt hier aan voldaan, omdat de persoonlijke sleutel k alleen bekend is bij de bankpas en de bank. Als de ontvangen transactiecode overeenkomt met de door de bank berekende transactiecode, dan kan de bank er van uitgaan dat de bankpas betrokken was bij het genereren van de code. Dit genereren gebeurt alleen als de juiste pincode is ingevoerd, waardoor het aantoonbaar is dat een transactie daadwerkelijk heeft plaatsgevonden. Of de bank de ontvangen transactiecodes opslaat om later nog aan te tonen dat een transactie heeft plaatsgevonden is onduidelijk. Jijzelf kan zonder hulp van de bank niet aantonen dat een transactie heeft plaatsgevonden. Alleen de bank en de kaart kennen namelijk de sleutel die is gebruikt.

3.2.2 ING bank

De achtereenvolgende handelingen die je bij de ING bank zelf moet uitvoeren om geld over te maken zijn te splitsen in twee mogelijkheden.

- Bij de keuze voor het bezitten van een lijst met TAN-codes geldt:
    1. De site geeft een nummer van de lijst;
    2. De TAN-code achter het gegeven nummer intypen op de site;
    3. Klik op Verzenden
- Bij de keuze voor het ontvangen van een TAN-code per sms geldt:
    1. Je ontvangt een TAN-code per sms (Transactie Autorisatie Nummer);

2. Typ de TAN-code in op de site;
3. Klik op Verzenden

Welke keuze je ook maakt, er wordt gebruik gemaakt van een TAN-code. Hoe deze TAN-code precies wordt gemaakt is niet bekend. Het kan zijn dat deze random worden gegenereerd. Hoe dan ook, als je in het bezit bent van een TAN-code door een lijst die rekening afhankelijk is of door het bezitten van de telefoon, dan ben je gerechtigd om in te loggen (*toegangscontrole*).
Bij de keuze voor een lijst is het belangrijk dat de lijst veilig wordt verstuurd naar de gebruiker. Dit gebeurt met de post. Bij de keuze voor het ontvangen van een TAN-code per sms, speelt telefoniebeveiliging een belangrijke rol. In beide gevallen wordt hiermee aan de beveiligingsaspecten *integriteit* en *privacy* voldaan.
Nadat de TAN-code is ingevoerd op de site en er op Verzenden is geklikt, wordt er gebruik gemaakt van een TLS verbinding. Hierdoor wordt *authenticatie* en *integriteit* aangetoond en blijft *privacy* behouden.
Vervolgens moet de bank de ontvangen TAN-code controleren. Omdat de bank weet wat de TAN-code moet zijn, kan zij deze dus direct verifiëren. Door het ontvangen van een juiste TAN-code kan de bank aantonen dat een transactie heeft plaatsgevonden (*verantwoording*).

3.2.3 ABN AMRO bank

De achtereenvolgende handelingen die je bij de ABN AMRO bank zelf moet uitvoeren om geld over te maken zijn:

1. Klik aan welke e.dentifier je hebt;
2. E.dentifier:
   (a) Pinpas invoeren in e.dentifier;
   (b) Druk op 2 Verzend opdr;
   (c) Pincode intypen;
   (d) Toets OK;
   (e) Typ de code van de site in op de e.dentifier;
   (f) Toets OK;
3. Response die op de e.dentifier verschijnt intypen op de site;
4. Klik op OK+Verzenden

Voor geldtransacties bij de ABN AMRO geldt hetzelfde als bij de Rabobank, behalve dat er geen tweede code wordt gegeven bij hoge bedragen.

Hoofdstuk 4

Cryptografie achter internetbankieren

In dit hoofdstuk zal de onderliggende wiskunde van de cryptografie, die gebruikt wordt bij internetbankieren, uitgelegd worden. Beginnend bij hashfuncties, gevolgd door DES (Data Encryption Standard), 3DES (Triple Data Encryption Standard) en RSA (Rivest Shamir Adleman).

4.1 Hashfuncties

Een cryptografische hashfunctie beeldt een input van willekeurige lengte af op een string van vaste lengte, welke de hashwaarde h wordt genoemd. Het is dus een functie $H : \mathbb{B}^* \to \mathbb{B}^k$. Hierbij is $\mathbb{B}^*$ een oneindige en $\mathbb{B}^k$ een eindige verzameling, waardoor er (vele!) X_1 en X_2 zijn waarvoor geldt: $X_1 \neq X_2$ en $H(X_1) = H(X_2)$. Met andere woorden, deze afbeelding is niet injectief.

Definitie 4.1. *Hashfunctie H is een éénrichtingsfunctie (one-way) als het ondoenlijk is uit de hashwaarde h een M te berekenen waarvoor geldt $H(M) = h$.*

Een hashfunctie met een hashwaarde van grootte 10 bits, kan in maximaal $2^{10} \approx 10^3$ pogingen gevonden worden. De keuze van de hashwaarde moet dus groot genoeg zijn om dit niet zomaar na te kunnen gaan. Vanaf ongeveer 100 bits is dit praktisch onmogelijk.

Definitie 4.2. *Hashfunctie H is zwak botsingsvrij als het, gegeven M, ondoenlijk is een $M' \neq M$ te vinden met $H(M) = H(M')$.*

Definitie 4.3. *Hashfunctie H is sterk botsingsvrij als het ondoenlijk is, berichten M_1 en M_2 te vinden zodat $M_1 \neq M_2$ en $H(M_1) = H(M_2)$.*

Stelling 4.1. *Zij H een hashfunctie, dan geldt: H sterk botsingsvrij $\Rightarrow$ H zwak botsingsvrij.*

Bewijs. Stel dat H niet zwak botsingsvrij is. Dan kan bij een gegeven M_1 een M_2 worden gevonden met dezelfde hashwaarde. Daardoor kan er, zonder kennis van een M_1 vooraf, een paar worden gevonden door M_1 random te kiezen en M_2 erbij te zoeken met dezelfde hashwaarde. Dus H is niet sterk botsingsvrij. Dus als H sterk botsingsvrij is dan is H zwak botsingsvrij. □

Stelling 4.2. *Zij H een hashfunctie, dan geldt: H zwak botsingsvrij $\Rightarrow$ H éénrichtingsfunctie (one-way).*

Bewijs. Veronderstel dat H geen éénrichtingsfunctie is. Dan kan bij een gegeven hashwaarde h een M worden gevonden met $H(M) = h$. Maar dan kan bij een gegeven (willekeurig) bericht M_1 eerst de hashwaarde $H(M_1)$ worden berekend en vervolgens wordt een M berekend met $H(M) = H(M_1)$. De kans dat M gelijk is aan M_1 is klein vanwege het grote aantal berichten met hashwaarde $H(M_1)$. Er is dus een tweede bericht gevonden met dezelfde hashwaarde, waarmee H niet zwak botsingsvrij is. Dus als H zwak botsingsvrij is dan is H een éénrichtingsfunctie. □

Deze eigenschappen zijn toe te passen om aan verschillende beveiligingsaspecten te voldoen.

1. Een hashfunctie moet een *éénrichtingsfunctie* zijn voor het behoud van *authenticatie*. Als je een bericht voorziet van je (persoonlijke) handtekening en hier vervolgens de hashwaarde van berekent, dan is het voor een ander ondoenlijk om een ander bericht te vinden met dezelfde hashwaarde. Met andere woorden, niemand kan de handtekening onder het bericht veranderen (zonder de hashwaarde te veranderen) en daarmee bewijs je dus dat jij de handtekening hebt gezet.

2. Een hashfunctie moet *zwak botsingsvrij* zijn voor het behoud van *integriteit* tegen een passieve aanval. Als een hashfunctie niet zwak botsingsvrij is, dan is het mogelijk om bij een gegeven bericht M een bericht M' te vinden met $H(M) = H(M')$. Hierdoor kan iemand die het bericht onderschept het bericht veranderen, terwijl het dezelfde hash-waarde heeft.

3. Een hashfunctie moet *sterk botsingsvrij* zijn voor het behoud van *integriteit* tegen een actieve aanval. Als een hashfunctie niet sterk botsingsvrij is, is het mogelijk om een bericht naar iemand anders te sturen met de vraag hier een handtekening onder te zetten. De ander stuurt

het ondertekende bericht terug, waarvan jij het bericht dat bij de handtekening staat, kan veranderen.

Bekende hashfuncties die in de praktijk worden gebruikt zijn: MD2 (Message-Digest Algorithm 2), MD4, MD5 en SHA-1 (Secure Hash Algorithm 1). Bij de beveiliging van internetbankieren wordt gebruik gemaakt van SHA-1 (zie Hoofdstuk 3). SHA-1 levert een hashwaarde op van 160 bits uit een bericht dat een maximumgrootte kan hebben van 2^{64} bits.

4.2 Data Encryptie Standaard

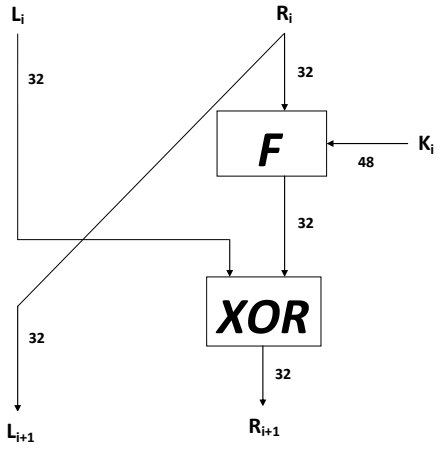
Het symmetrisch algoritme DES (Data Encryption Standard) is ontwikkeld in de jaren 70. Het kende vele toepassingen, totdat er een aanval op bekend werd. De opvolger van DES staat bekend als 3DES (triple DES). Dit algoritme wordt op dit moment nog altijd veel gebruikt, onder andere bij het internetbankieren.
Hieronder staat beschreven hoe DES werkt en waardoor het niet meer toepasbaar is. Daarna wordt de werking van 3DES beschreven aan de hand van de werking van DES.

4.2.1 DES

DES is een algoritme dat werkt op blokken klare tekst van 64 *bits*, waarbij de data in 19 achtereenvolgende ronden wordt geëncrypt.

In ronde 1 en 19 vindt een *initiële permutatie* (IP) plaats, waarbij de eindpermutatie de inverse is van de initiële permutatie (IP^{-1}). Hierdoor voegt het niets toe aan de veiligheid en heeft het ook geen invloed bij het decrypten, maar het enige doel is om eventuele concentraties enen en nullen te spreiden. Ronde 2 tot en met 17 wordt *DES-iteratie* gebruikt. In ronde 18 worden de linker- en de rechthelft van de data verwisseld, wat er voor zorgt dat de decryptie goed gaat.

Bij de *DES-iteratie* wordt in elke ronde de helft van de data versleuteld, waarbij sleutels worden gebruikt van 48 bits die afgeleid zijn van de 56 sleutelbits. Deze versleuteling gebeurt volgens onderstaande figuur:



Definitie 4.4. *De* XOR-functie *staat voor een $\oplus$-operatie. Deze functie beeldt twee bits af op 0 als ze gelijk zijn en op 1 als ze ongelijk zijn. Deze functie heeft als bijzondere eigenschap dat de operatie zijn eigen inverse is.*

Allereerst wordt de data van 64 gesplitst in een linker- en een rechterhelft (L_i en R_i) van ieder 32 bits. Vervolgens geldt (zoals te zien is in de figuur hierboven):

$$L_{i+1} = R_i$$

$$R_{i+1} = L_i \oplus F(R_i, K_i)$$

Deze iteratie staat ook wel bekend als een *Feistel-netwerk*.

Het *decrypten* is de methode om bij gegeven L_{i+1} en R_{i+1}, de L_i en R_i te weten te komen. Hiervoor geldt:

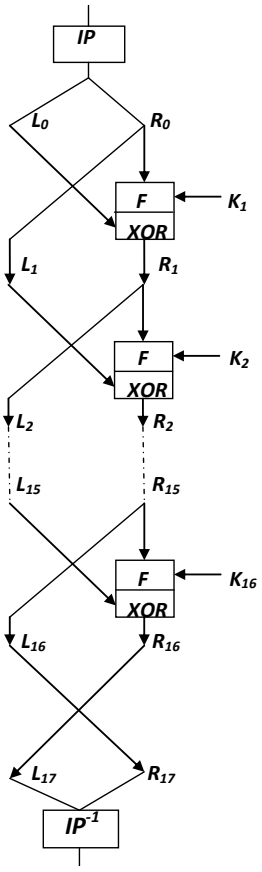
$$L_{i+1} = R_i \tag{4.1}$$

$$R_{i+1} \oplus F(L_{i+1}, K_i) = L_i \oplus F(R_i, K_i) \oplus F(R_i, F_i) = L_i, \tag{4.2}$$

want $R_{i+1} = L_i \oplus F(R_i, K_i)$ en de iteratie is zijn eigen inverse.

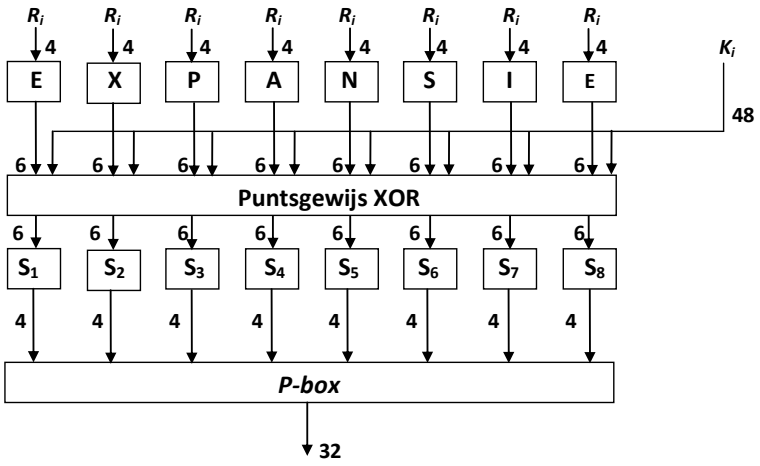
Met andere woorden, het complete proces wordt geïnverteerd door de uitvoer weer bovenin in te voeren, maar met de iteratiesleutels in omgekeerde volgorde. Op deze manier worden niet alleen de twee datahelften precies teruggevonden, maar ook hun volgorde is alvast verwisseld voor het inverteren van de volgende iteratie.

Het gehele algoritme ziet er dus als volgt uit:



Hieronder is verder toegelicht wat de functie F doet en vervolgens hoe de sleutel K_i wordt gegenereerd.

De meeste cryptografie zit in de functie F. Deze kan als volgt grafisch weergegeven worden:



Allereerst wordt de rechterhelft van de data R_i en de iteratie-sleutel K_i in acht blokjes verdeeld, waardoor in elk blokje van R_i vier bits zitten en in elk blokje van K_i zes bits zitten. Om dit puntsgewijs te kunnen XOR-en moet elk blokje van R_i er nog twee bits bij krijgen. Dit gebeurt door middel van de *expansie-functie*.

Definitie 4.5. *Een* expansie-functie *geeft elk blokje een bit van het linker-buurblokje en een van het rechterbuurblokje (waarbij de twee uiteinden ook als buren worden beschouwd).*

De geXORde blokken, die ontstaan na het toepassen van de *XOR-functie*, zijn nu de input van een substitutie-box, ookwel *S-box* genoemd. De acht S-boxen zijn verschillend, maar in elk van de rondes 1 tot en met 17 zijn de acht S-boxen wel gelijk. Afhankelijk van het blokje $c_1c_2c_3c_4c_5c_6$ krijgt de S-box een output van vier bits. Deze vier bits zijn een substitutie van $c_2c_3c_4c_5$, waarbij de substitutiepermutatie wordt gekozen door c_1 en c_6. Een voorbeeld van een S-box zie je hieronder:

	$c_2c_3c_4c_5$															
c_1c_6	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	1110	0100	1101	0001	0010	1111	1011	1000	0011	1010	0110	1100	0101	1001	0000	0111
01	0000	1111	0111	0100	1110	0010	1101	0001	1010	0110	1100	1011	1001	0101	0011	1000
10	0100	0001	1110	1000	1101	0110	0010	1011	1111	1100	1001	0111	0011	1010	0101	0000
11	1111	1100	1000	0010	0100	1001	0001	0111	0101	1011	0011	1110	1010	0000	0110	1101

Als laatste worden de 32 bits, de output van de S-boxen, gepermuteerd in een *P-box*. Het doel van deze permutatie is dat elke S-box in de volgende ronde door meerdere S-boxen in vorige rondes wordt beïnvloed.

Zoals eerder genoemd bevatten de sleutels 56 bits, terwijl ze worden ingevoerd als 64 sleutelbits. Van deze 64 bits is elke achtste bit een *pariteitsbit* is die de juistheid van de zeven bits ervoor controleert. Deze acht *pariteitsbits* doen in het algoritme dus niet mee.
Uit deze 56 bits worden er steeds 48 gebruikt voor een iteratie. Deze 48 bits worden bepaald door allereerst de 56 bits te verdelen over twee 28-bitsbrede schuifregisters. Deze *sleutelinvoerselectie* bestaat uit vier keer zeven *bytes*. Vervolgens neemt men links de eerste bit van elke byte in aflopende volgorde, daarna de tweede etc. Rechts begint bij de zevende bit, daarna de zesde etc. en aan het eind de vier vierde bits. Daarna wordt in elke ronde een *sleutelrotatie* naar links toegepast over een of twee bits volgens onderstaande tabel:

Ronde	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Verschuiving	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

Als laatste vindt *sleutelcompressie* plaats waarmee 48 bits worden geselecteerd.

Voor het decrypten is het handig dat er 28 verschuivingen plaatsvinden, dit is in te zien door het aantal verschuivingen in bovenstaande tabel bij elkaar op te tellen. Dit betekent dat de 16^e iteratie-sleutel bestaat de *sleutelcompressie* losgelaten op de *sleutelinvoerselectie*. Hiermee zijn de voorgaande iteratie-sleutels eenvoudig te vinden, wat handig is voor de decryptie. Het enige verschil met encryptie is dat we hierbij met één of twee posities naar rechts schuiven, waarbij de tabel van achteren naar voren wordt afgelezen.

Nu we weten hoe DES precies werkt, bekijken we waardoor het tegenwoordig niet meer wordt gebruikt. Het nadeel van DES is de sleutellengte van 56 bits, die met een *brutekracht-aanval* te breken is.

Definitie 4.6. *Een* brutekracht-aanval *is een aanval die de bits kan achterhalen door simpelweg alle sleutels uit te proberen totdat er een leesbaar resultaat ontstaat.*

In het geval van DES zijn dit 2^{56} mogelijke sleutels. Deze aanval was al lange tijd bekend, terwijl er pas in 1998 een machine door Electronic Frontier Foundation hiervoor werd gepresenteerd. Hierdoor wordt DES nu niet meer gebruikt. Toch kunnen we zeggen dat DES algoritmisch zeer sterk is, maar door zijn korte sleutels niet voldoende weerstand biedt tegen een brute kracht aanval.

Om deze aanval tegen te gaan werd 2DES (double DES) bedacht. Hierbij maakt men gebruik van twee sleutels. Echter ontdekte men dat het rekenwerk voor een *meet-in-the-middle aanval* in dezelfde orde is als een *brutekracht-aanval* op DES.

Definitie 4.7. *Een* meet-in-the-middle aanval *is een aanval waarbij je met behulp van de klare tekst en de cijfertekst, de sleutels kunt vinden.*

Toch is een aanval op 2DES waarschijnlijk niet geheel uitvoerbaar vanwege het geheugengebruik. Door deze onzekerheid wordt 2DES niet gebruikt en beschrijven we dit algoritme hier ook niet.

4.2.2 3DES

Triple DES is ontwikkeld om de sleutellengte en de veiligheid van DES te vergroten. Dit gebeurt door DES drievoudig toe te passen, waarbij drie

onafhankelijke sleutels worden afgesproken. Dit kan op twee verschillende manieren:

- Twee verschillende sleutels van 56 bits waarbij de eerste en derde bewerking met dezelfde sleutel wordt uitgevoerd en waarbij effectief een sleutel van 112 bits ontstaat.
- Drie verschillende sleutels van 56 bits waarbij effectief een sleutel van 168 bits ontstaat.

Bij 3DES wordt de encryptie met de sleutels (k_1, k_2, k_3) toegepast op een bericht M door:

$$E_{k_3}(D_{k_2}(E_{k_1}(M))) = E_{3DES}(M)$$

Decryptie op $E_{3DES}(M)$ gaat dan als volgt:

$$D_{k_1}(E_{k_2}(D_{k_3}(E_{3DES}(M)))) = M$$

Er zijn nog geen aanvallen op 3DES bekend. De brutekracht-aanval die kon worden toegepast op DES is hier ondoenlijk, omdat hier geen sprake is van 2^{56}, maar 2^{112} of 2^{168} mogelijke sleutels. Daarom heeft 3DES nog altijd belangrijke toepassingen.

4.3 TLS

TLS staat voor Transport Layer Security en is de opvolger van SSL (Secure Sockets Layer). TLS zorgt dat er op het internet veilig gecommuniceerd wordt. Bij het internetbankieren zorgt TLS voor een veilige verbinding met de server van de bank.

Zodra A aangeeft dat hij wil communiceren met een server over een beveiligde verbinding, dus nadat A op Inloggen heeft geklikt, start het *handshake protocol*:

1. A stuurt een bericht met een random getal en informatie over ondersteunde algoritmes en protocollen naar de server van de bank. Daarnaast vraagt A nog naar de public key en het certificaat van de server.
2. De server stuurt de keuze van het algoritme, de keuze van het protocol, een random getal, de public key en het certificaat terug. Ook zal de server naar de public key en het certificaat vragen van A.
3. A stuurt de public key, het certificaat en de sleutelgeneratie.

Door middel van de certificaten wordt *authenticatie* aangetoond. Een certificaat is alleen te krijgen bij de Certification Authority (CA) als is aangetoond dat de verbinding tussen een sleutel en een persoon gerechtvaardigd is. In een certificaat staat een nummer, de uitgever, een ontvanger, een geldigheid, een public key en een handtekening van de CA voor alle gegevens op het certificaat.
De random getallen die worden meegestuurd zijn er onder andere voor om te zorgen dat er meerdere SSL-protocollen naast elkaar kunnen lopen zonder in de war te raken. Bij sommige algoritmes worden ze ook voor de sleutelgeneratie gebruikt en tevens worden ze bij de *MAC* (Message Authentication Code) gebruikt.
Sleutelgeneratie kan door middel van zeer veel protocollen, maar in de meeste gevallen wordt er gekozen voor *RSA* (zie hoofdstuk 4). Dit komt doordat RSA massaal wordt ondersteund en relatief snel kan zorgen voor lange sleutels.

Als het handshake protocol tot een goed einde is gebracht, wordt het bericht met behulp van de gegenereerde sleutels en symmetrische cryptografie versleuteld en naar de tegenpartij verzonden. Hierdoor blijft de *privacy* behouden. Bij deze versleuteling wordt vaak gebruik gemaakt van het symmetrische algoritme 3DES.
Ieder bericht dat wordt verstuurd is voorzien van een *MAC*, de berichtlengte en een beknopte samenvatting van de inhoud.
De MAC is een hashfunctie met een geheime sleutel als input. Deze hashfunctie zorgt voor *integriteit*. Bij het internetbankieren wordt hiervoor SHA-1 gebruikt. Is de MAC onjuist, dan stopt de communicatie direct (Alert protocol). Willen de partijen toch nog veilig communiceren, dan dient de hele sessie opnieuw te worden gestart.

4.4 RSA

RSA is een asymmetrisch cryptografisch systeem, waarbij dus één sleutel openbaar is en de andere geheim. Het is bedacht door R.L. Rivest, A. Shamir en L. Adleman, waar het ook naar vernoemd is.
Het RSA-systeem is gebaseerd op het feit dat het eenvoudig is om het product van twee grote priemgetallen te berekenen, maar het erg moeilijk is om uit dit product de oorspronkelijke priemgetallen te bepalen.

RSA werkt als volgt:
Allereerst moeten er twee grote priemgetallen, p en q, worden gegenereerd.

Van deze twee priemgetallen wordt het product berekend, $n = p \cdot q$. Vervolgens wordt een getal e bepaald, waarvoor geldt:

$$3 < e < (p-1)(q-1).$$

Tevens moet gelden:

$$\text{ggd}(e, (p-1)(q-1)) = 1.$$

Dat betekent dat e en $(p-1)(q-1)$ *relatief priem* zijn (ookwel *copriem* genoemd). Met behulp van het algoritme van Euclides wordt d berekend zodat:

$$e \cdot d \equiv 1 \bmod (p-1)(q-1). \tag{4.3}$$

Nu is de openbare sleutel (e, n), terwijl p, q en d geheim blijven.

Het versleutelen van een bericht werkt nu als volgt:
Bericht M (waarvan we aannemen dat het een decimaal geschreven getal is) wordt opgedeeld in blokken B_i. De blokken worden versleuteld volgens de formule:

$$C_i \equiv B_i^e \bmod n.$$

Alle blokken worden nu weer achter elkaar gezet en vormen de cijfertekst C.

Om het bericht te ontcijferen moet met elk blok de volgende bewerking worden uitgevoerd:

$$B_i \equiv C_i^d \bmod n.$$

Nu geldt:

$$C_i^d \equiv (B_i^e)^d \equiv B_i^{ed} \bmod n.$$

Omdat $e \cdot d \equiv 1 \bmod (p-1)(q-1)$, volgt hieruit dat voor zekere $k \in \mathbb{N}$ geldt:

$$B_i^{ed} = B_i^{k\cdot(p-1)(q-1)+1} = B_i \cdot B_i^{k\cdot(p-1)(q-1)} \tag{4.4}$$

$$B_i \cdot B_i^{k\cdot(p-1)(q-1)} \equiv B_i \cdot 1 \equiv B_i \bmod n. \tag{4.5}$$

In vergelijking 4.5 maken we gebruik van de Stelling van Euler.

Stelling 4.3 (Stelling van Euler)**.** *Als a en n positieve gehele getallen zijn waarvoor geldt dat ze relatief priem zijn, dan geldt:*

$$a^{\phi(n)} \equiv 1 \bmod n.$$

Hierbij is $\phi(n)$ de indicator van n, gedefinieerd als $\phi(n) = n \cdot \prod_{p|n}(1 - \frac{1}{p})$. Dit komt overeen met het aantal positieve gehele getallen, kleiner dan n, dat geen factor gemeen heeft met n.

Uit deze stelling volgt direct vergelijking 4.5, want $\phi(n) = \phi(p \cdot q) = \phi(p) \cdot \phi(q) = (p-1)(q-1)$ met p en q priem.
Net als bij het vercijferen zet je alle blokken weer achter elkaar, zodat het gehele bericht te lezen is.

Stelling 4.4 (Hulpstelling)**.** *Laat n een positief geheel getal zijn en ggd$(a, n) = 1$, dan geldt:*

$$\textit{Als } a \cdot i \equiv a \cdot j \bmod n, \textit{ dan } i \equiv j \bmod n$$

Bewijs van de hulpstelling. $a \cdot i \equiv a \cdot j \bmod n$ betekent dat $a \cdot i = a \cdot j + k \cdot n$ voor een zekere $k \in \mathbb{N}$. Met andere woorden: $n \mid a \cdot i - a \cdot j$.
Dus: $n \mid a \cdot (i - j)$. Maar ggd$(a, n) = 1$, dus $n \mid i - j \Rightarrow i \equiv j \bmod n$ □

Bewijs Stelling van Euler. Beschouw de getallen die geen factor met n gemeen hebben als $b_1, b_2, \ldots, b_m$ met voor alle m; $b_m < n$.
Vermenigvuldig deze getallen a zodat $a \cdot b_1, a \cdot b_2, \ldots, a \cdot b_m$ ontstaat. Volgens de hulpstelling zijn deze getallen verschillend. Er geldt dan:

$$a \cdot b_1 \cdot a \cdot b_2 \cdot \ldots \cdot a \cdot b_m \equiv b_1 \cdot b_2 \cdot \ldots \cdot b_m \bmod n.$$

Delen door $b_1 \cdot b_2 \cdot \ldots \cdot b_m$ geeft: $a^m \equiv 1 \bmod n$. Hierin is m het aantal getallen, kleiner dan n, dat *relatief priem* is met n. Dit komt overeen met de definitie voor $\phi(n)$. Ofwel: $a^{\phi(n)} \equiv 1 \bmod n$. □

De veiligheid van RSA is te danken aan het factoriseren van twee priemgetallen p en q uit n. Dit is ondoenlijk als p en q erg groot zijn.
Als we $\phi(n)$ kunnen berekenen (met $n = p \cdot q$), dan kunnen we p en q berekenen, want $\phi(n) = \phi(p) \cdot \phi(q) = (p-1)(q-1) = pq - (p+q) + 1$. Hierin is $pq = n$ openbaar, zodat we $p + q$ kunnen berekenen. Hieruit volgen twee vergelijkingen met twee onbekenden, die we kunnen oplossen. Als we p en q kunnen vinden, dan zou je met behulp van de openbare sleutel (e, n) de decryptiesleutel d kunnen bepalen, zie vergelijking 4.3. We gaan er van uit dat het berekenen van $\phi(n)$ even moeilijk is als het factoriseren van n. Als $\phi(n)$ niet slecht gekozen is, dan zal het moeilijk zijn om d te vinden zonder $\phi(n)$. Dankzij deze veiligheid beschikt alleen diegene die de openbare sleutel (e, n) uitgegeven heeft over d, waardoor hij als enige de mogelijkheid heeft om cijferteksten te ontcijferen.

Bibliografie

[1] Beutelspacher, A. (2002). *Mathematics everywhere*, tweede editie, hoofdstuk 10.

[2] Bosma, W. (2011). *Cursus cryptografie*.

[3] *Digipass Family of Authentication Devices*

[4] Driessens, E. *Security op het web*.

[5] Lubbe, J.C.A. van der (1994). *Basismethoden Cryptografie*, blz. 140-152.

[6] Lubbe, J.C.A. van der (1997). *Basismethoden Cryptografie*, tweede druk, paragraaf 1.1, 1.2 en 6.2.

[7] Schneier, B. *Applied cryptography*, tweede editie, hoofdstuk 12.

[8] Tel, G. (2006). *Cryptografie, beveiliging van de digitale maatschappij*, paragraaf 1.1, 2.1, 2.2, 3.2, 3.4, 5.3, 6.3, 7.2, 9.3 en 12.1.

[9] Weger, B. de (2003). *Getaltheorie die gebruikt wordt in RSA*.