Knoeiboelfuncties

Benne de Weger

Bij "Cryptografie" denk je waarschijnlijk allereerst aan versleutelen: het omvormen van een leesbare tekst tot een geheimschrift. Dat is niet zo gek, de geschiedenis **van de cryptografie is lang** de geschiedenis geweest van geheimschriftmethoden.

Sinds enkele tientallen jaren is cryptografie veel breder geworden. Met sleutels kun je meer dan geheimschriften maken. Je kunt er deuren mee openen, d.w.z. toegang krijgen tot voorzieningen, zoals het gebruik van een computer (sleutel = wachtwoord) of van je pinpas (sleutel = pincode).

Een steeds belangrijker wordende toepassing is de digitale handtekening. Zoals een papieren document van een gewone handtekening voorzien kan worden, zo kan bij een "digitaal document" met behulp van een cryptografische sleutel een digitale handtekening gemaakt worden. Het document wordt verbonden aan jouw persoon: jij bent de enige die die handtekening kan maken, jij bent de enige die de juiste cryptografische sleutel heeft. In beide gevallen heeft iemand anders wel de mogelijkheid die handtekening te controleren op echtheid.

Het maken van een digitale handtekening is een nogal dure cryptografische handeling. Voor een bestand van een paar MB's gaat dat al gauw seconden duren, en dat is gewoon te lang. Onder andere daarom is er een nieuw soort cryptografische functie bedacht, de *hashfunctie*[1]. De laatste jaren ben ik daar in geïnteresseerd geraakt, en wij hebben in Eindhoven op dat gebied aardige resultaten geboekt, voor een belangrijk deel dankzij mijn afstudeerder Marc Stevens.

Je kunt een hashfunctie definieren als een functie $h:\{0,1\}^* \to \{0,1\}^n$. Dat wil zeggen: je kan er een willekeurige rij bits instoppen, van willekeurige lengte, en er komt weer een rij bits uit, maar dan van vaste lengte **n**. Een hashfunctie is cryptografisch sterk als ze voldoet aan de volgende eigenschappen:

• éénwegfunctie: het is makkelijk om, gegeven x, de hash $h(x)$ te berekenen, maar het is praktisch gezien ondoenlijk om de functie te inverteren, dus om, gegeven y, een x te vinden die hash y heeft, dus zodat $y = h(x)$.

• botsingbestendig: het is praktisch gezien ondoenlijk om een botsing te vinden, d.w.z. om twee verschillende x_1, x_2 te vinden met dezelfde hash, dus $h(x_1) = h(x_2)$.

De echte wiskundige zegt nu meteen: er zijn oneindig veel mogelijke x-jes en maar 2^n mogelijke waarden voor $h(x)$, dus natuurlijk zijn er botsingen. Dat klopt. Het punt is alleen: bij een goede hashfunctie kun je die pas na $2^{n/2}$ berekeningen vinden. En als je dan n maar groot kiest, bv. 160, dan lukt niemand dat.

Hoe een hashfunctie precies werkt is lastig uit te leggen, en verschilt ook sterk per hashfunctie. Houd het er maar op dat een hashfunctie inderdaad een grote knoeiboel van de invoer-bits maakt, waar als het goed is niets meer van de invoer in te herkennen is.

Het grote voordeel van hashfuncties is dat ze heel snel grote hoeveelheden invoer kunnen verwerken. Honderden MB per seconde is geen probleem. In plaats van een groot document onderteken je alleen de hash, dat kan wel binnen een aanvaard-

[1] *Het woord hash heeft hier niets met drugs te maken. Van Dale's Handwoordenboek E-N geeft als meest geschikte vertaling mengelmoes. Vergelijk hachee. De zin make a hash of it betekent de boel verknoeien. Automatische vertaalprogramma's vertalen hash function met knoeiboelfunctie.*

bare tijd, inclusief het berekenen van de hash. Nu wordt ook duidelijk waarom een hashfunctie botsingbestendig moet zijn: als je twee verschillende documenten met dezelfde hash zou kunnen maken, dan kun je de ene laten ondertekenen door je baas, en die handtekening is dan meteen ook geldig voor het andere document. En dat is natuurlijk niet de bedoeling.

Een andere interessante toepassing van hashfuncties is als *commitment*. Je kunt daarmee vastleggen dat je een bepaald document hebt, zonder dat je dat document zelf hoeft te laten zien. Zet bijvoorbeeld de zin "Nederland wordt in 2008 wereldkampioen" in een bestand, bereken de hash van dat bestand, en publiceer die hash in een advertentie in de krant. Als Nederland volgend jaar inderdaad wereldkampioen is geworden, dan maak je het bestand ook openbaar, en iedereen zal onder de indruk zijn van jouw voorspellende gave. Als Nederland geen wereldkampioen wordt, dan heb je een probleem (maar ja, dan heeft heel Nederland een probleem…)

Eén van de meest gebruikte hashfuncties is MD5. In augustus 2004 werd de cryptografische wereld opgeschrikt door een relatief onbekende Chinese mevrouw, Xiaoyun Wang, die op een grote conferentie plotseling een botsing voor MD5 liet zien.

De botsingen die haar methode oplevert zien er nogal random uit: het zijn geen leesbare teksten. Sinds Wang's eerste bekendmaking zijn er verschillende groepen, waaronder wij, in geslaagd om toch zinvolle documenten te maken die botsen onder MD5. Je moet zo'n random uitziend blok bits dan op een of andere manier inbouwen in een document. De botsingen van Wang hebben de prettige eigenschap dat de botsing-eigenschap behouden kan blijven als je er zelfgekozen bits vooraan en achteraan plakt. Die bits kun je zinvolle teksten laten zijn, en zo kun je bijvoorbeeld de botsende bits verstoppen in een plaatje, waar wat random bits wellicht niet opvallen. Er zijn allerlei andere slimme dingen bedacht om botsingen in te bouwen in zinvolle documenten. Maar een groot nadeel van de botsingen van Wang is dat de tekst die je ervoor en erachter plakt, voor beide botsende bestanden exact hetzelfde moet zijn. De twee bestanden mogen alleen in de botsende bit-blokken zelf verschillen.

Marc Stevens' afstudeerwerk ging over de vraag of dat ook beter kan. Het is hem gelukt de methode van Wang zo uit te breiden dat je voorafgaand aan een botsing nu willekeurige teksten kunt plakken, voor beide botsende bestanden verschillend, en helemaal zelf gekozen, terwijl toch de botsing-eigenschap behouden blijft. Dat is een flinke doorbraak, waarvoor Marc al beloond is met het mogen houden van de openingsvoordracht op de grote cryptografie-conferentie EuroCrypt, op 21 mei in Barcelona. Ik wil tenslotte laten zien hoe je hiermee geld kunt verdienen ☺.

Maak drie bestanden met de volgende teksten:

m1 = "Nederland wordt in 2008 wereldkampioen"
m2 = "Duitsland wordt in 2008 wereldkampioen"
m3 = "Nederland noch Duitsland wordt in 2008 wereldkampioen"

Vraag Marc of hij een botsing voor $\mathbf{m}_1$ en $\mathbf{m}_2$ wil maken, dat wil zeggen, rijtjes bits $\mathbf{b}_1$ en $\mathbf{b}_2$ zodat $\mathbf{MD5(m_1,b_1) = MD5(m_2,b_2)}$. Dat kan Marc, maar of hij het wil hangt er natuurlijk vanaf hoeveel het schuift. Vraag Marc (tja, het gaat je wel een investerinkje kosten) vervolgens of hij een botsing kan maken waardoor deze twee gebotste berichten met de derde gaan botsen, dat betekent: rijtjes bits $\mathbf{c}_{12}$ en $\mathbf{c}_3$ zodat $\mathbf{H = MD5(m_1,b_1,c_{12}) = MD5(m_2,b_2,c_{12}) = MD5(m_3,c_3)}$.

Dat kan dan ook nog allemaal zo dat die rare rijtjes extra bits netjes verstopt kunnen worden in een heel klein plaatje. Nu publiceer je de hashwaarde H in de krant, en je gaat met al je vrienden en bekenden weddenschappen aan dat jij heel goed kunt voorspellen wie volgend jaar wereldkampioen wordt. Wint Nederland, dan laat je bericht $\mathbf{m}_1,\mathbf{b}_1,\mathbf{c}_{12}$ zien, wint Duitsland dan bericht $\mathbf{m}_2,\mathbf{b}_2,\mathbf{c}_{12}$ en als een ander land wint, bericht $\mathbf{m}_3,\mathbf{c}_3$. De hash $\mathbf{H}$ klopt altijd. Veel succes.

Benne de Weger