De situatie van MD5 en SHA-1

Hash-functies onder vuur

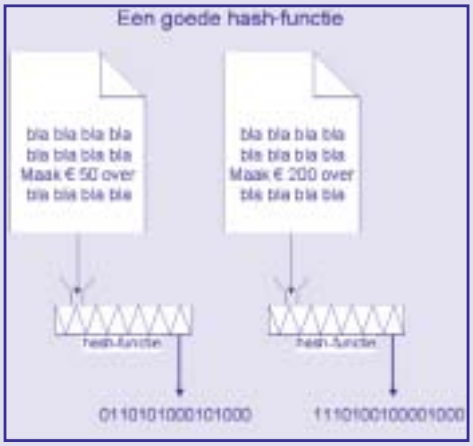
Auteur: Benne de Weger > Benne (dr B.M.M.) de Weger is universitair docent in de cryptologie-groep van de Faculteit Wiskunde en Informatica van de Technische Universiteit Eindhoven, e-mail: b.m.m.d.weger@tue.nl; web: http://www.win.tue.nl/~bdeweger.

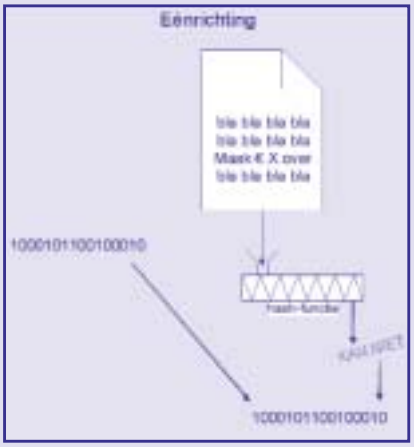
Wat is een hash-functie?

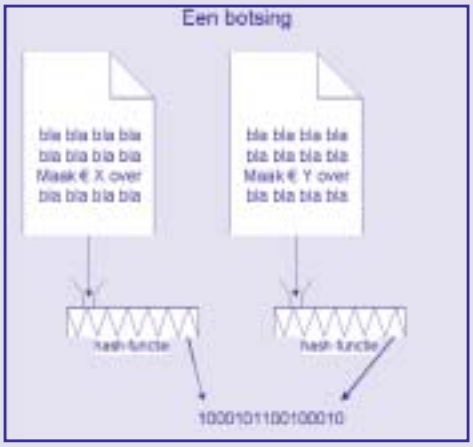
Een cryptografische *hash-functie* is een functie die als input een willekeurige bitstring (rij bits, bijvoorbeeld een elektronisch document of een computerprogramma) accepteert en hiervan een *vingerafdruk* maakt. Zo'n vingerafdruk (hierna noemen we die een *hash-waarde*; de term *message digest* wordt ook wel gebruikt) is een rijtje bits met een vaste lengte, bijvoorbeeld 128 bits voor MD5, of 160 bits voor SHA-1. Twee verschillende inputs die dezelfde hash-waarde hebben worden een *botsing* genoemd en dat moet vermeden worden.

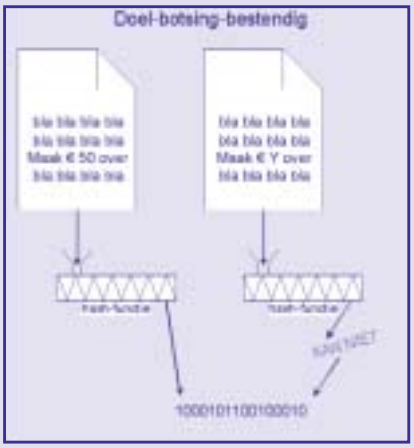
MD5 gekraakt! SHA-1 ook?

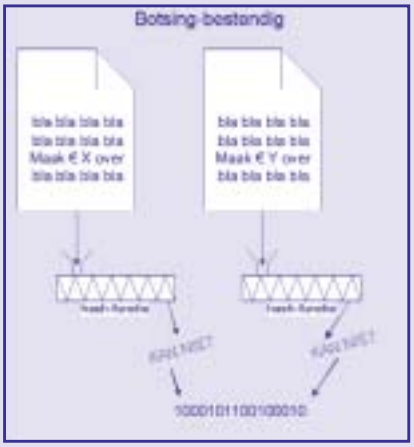
In augustus 2004 stond de crypto-wereld op z'n kop. Op de conferentie *Crypto 2004* in Santa Barbara, Californië, werd bekendgemaakt dat er botsingen gevonden zijn voor enkele cryptografische hash-functies, waaronder het veel gebruikte MD5. Daarmee heet MD5 gekraakt te zijn. Ook kwam de eveneens veelgebruikte en als veilig bekendstaande hash-functie SHA-1 onder vuur te liggen. In februari 2005 werd bekend dat SHA-1 inderdaad zwakker is dan cryptologen altijd dachten. Deze gebeurtenissen hebben niet alleen in de gemeenschap van academische cryptologen veel opzien gebaard, ook de gebruikers van hash-functies, in toepassingen zoals digitale handtekeningen, werden opgeschrikt en vragen zich af wat de consequenties zijn voor de veiligheid van hun toepassingen. Het leidende Amerikaanse standaardiseringsinstituut NIST (National Institute of Standards and Technology) heeft in augustus 2004, vlak na het bekend worden van het nieuws over MD5, al geconcludeerd dat voor gebruik door de Amerikaanse overheid SHA-1 vervangen moet gaan worden, uiterlijk per 2010. In dit artikel willen we nagaan, zonder in al te veel technische details te treden, wat er precies aan de hand is, wat voor gevolgen dat heeft, en hoe gebruikers het beste met deze situatie kunnen omgaan.











Een goede cryptografische hash-functie heeft in ieder geval de volgende eigenschappen:

- *één-richting (one-way, pre-image resistant*): het is praktisch gezien onmogelijk om uit een gegeven waarde een input-bitstring af te leiden met de gegeven waarde als hash-waarde;
- *doel-botsing-bestendig (target collision resistant, second pre-image resistant):* het is praktisch gezien onmogelijk om uit een gegeven input-bitstring een tweede input-bitstring af te leiden met dezelfde hash-waarde;
- *botsing-bestendig (random collision resistant)*: het is praktisch gezien onmogelijk om twee input-bitstrings te vinden met dezelfde hash-waarde.

Er zit een volgorde in deze drie eigenschappen, van sterk naar minder sterk:
Als een hash-functie niet één-richting is, dan is hij zeker niet doel-botsing-bestendig. Bij een gegeven input m_1 kan h = hash(m_1) berekend worden; gebruik dan het ontbreken van de één-richting-eigenschap om m_2 te vinden met hash(m_2) = h = hash(m_1).
Als een hash-functie niet doel-botsing-bestendig is, dan is hij zeker niet botsing-bestendig. Kies maar een willekeurige input m_1 en gebruik het ontbreken van de doel-botsing-bestendigheid om een input m_2 te vinden met dezelfde hash-waarde).

Iedere hash-functie heeft in beginsel botsingen. Er zijn immers oneindig veel inputs mogelijk en maar eindig veel hash-waarden, er zullen dus heel veel botsingen zijn. Maar dit is alleen een theoretische zorg. Het punt is namelijk dat voor een cryptografisch sterke hash-functie botsingen slechts te berekenen of te construeren zijn met een astronomische hoeveelheid rekenkracht en -tijd, zoveel dat dat praktisch gezien ondoenlijk is.

Deze drie eigenschappen onderscheiden een cryptografische hash-functie van een checksum, zoals CRC16 of CRC32. Dat CRCs geen sterke hash-functies zijn, komt alleen al door hun korte hash-lengte (16 resp. 32). Het vinden van botsingen en zelfs het vinden van pre-images, is daardoor erg eenvoudig. Maar ook afgezien van de lengte zijn ze cryptografisch gezien erg zwak.

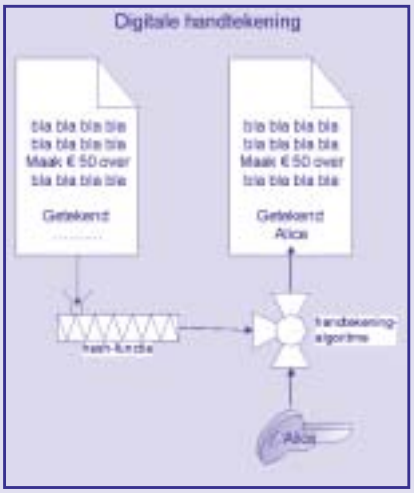
Hoe wordt een hash-functie gebruikt?

Zonder uitputtend te willen zijn vermelden we de belangrijkste toepassingen van hash-functies. Een eerste gebruik van hash-functies is het beschermen van de *integriteit* van gegevens. Wordt van een document de hash-waarde berekend en apart opgeslagen, dan kan een verandering in het document (zelfs van één bit) later altijd gedetecteerd worden door opnieuw de hash-waarde te berekenen en te vergelijken met de opgeslagen waarde. Dit kan bijvoorbeeld gebruikt worden om *onbedoelde wijzigingen* in software (zoals infectie door een virus) of gegevens (bijvoorbeeld voor forensisch onderzoek) te detecteren.

In het geval van *opzettelijke wijzigingen* heeft dit gebruik van hash-functies alleen zin als de aanvaller geen controle heeft over de opgeslagen hash-waarde. Anders kan hij immers die referentie-hash-waarde eenvoudig mee-wijzigen, zodat de verificatie toch zal kloppen.

Een tweede gebruik van hash-functies is als essentieel onderdeel van een digitale handtekening-protocol op basis van asymmetrische cryptografie. Zo'n digitale handtekening wordt

berekend uit de hash-waarde van de te ondertekenen gegevens, bijvoorbeeld een document dat een contract bevat. Daarmee wordt de identiteit van de ondertekenaar onloochenbaar verbonden aan zo'n document, op een cryptografisch en juridisch sterke manier. De veiligheid daarvan is (onder meer) gebaseerd op de veronderstelling dat de hash-waarde gezien kan worden als een unieke identificatie voor die gegevens. Dit wordt gegarandeerd door de botsing-bestendigheid van de hash-functie.



In het bijzonder worden digitale handtekeningen gebruikt door *Certification Authorities* (CA's) die PKI-certificaten uitgeven. Een certificaat wordt digitaal ondertekend door de CA, die daarvoor een hash-functie gebruikt. Daarmee garandeert zo'n CA dat een bepaalde publieke sleutel toebehoort aan een bepaalde eigenaar.

De hash-functies die in de praktijk het meeste gebruikt worden zijn MD5 en SHA-1.

Waarom zijn botsingen ongewenst?

Voor het gebruik van een hash-functie voor bescherming van de integriteit van gegevens is doel-botsing-bestendigheid nodig: een aanvaller moet niet in staat zijn bij een gegeven document een ander document te maken met dezelfde hash-waarde; anders zal hij wijzigingen ongemerkt kunnen doorvoeren.
Voor het gebruik bij digitale handtekeningen is in eerste instantie ook doel-botsing-bestendigheid nodig: een aanvaller moet niet in staat zijn om een tweede document te maken met dezelfde hash als een ander document. Als hij dat wel zou kunnen, dan kan hij een digitale handtekening onder een document zonder probleem overzetten op een ander, vals document, en dan zou hij later kunnen claimen dat de ondertekenaar het vervalste document heeft ondertekend. Deze vervalsingen zijn niet te detecteren. In het geval van door een CA ondertekende certificaten kan de koppeling van de identiteit van een persoon aan een publieke sleutel niet meer gegarandeerd worden.
In tweede instantie echter is meestal zelfs botsing-bestendigheid vereist. Dat is zeker zo als de aanvaller zelf de beide inputs voor de hash kan bepalen.

Hoe weten cryptologen of een hash-functie goed is?

Het enigszins teleurstellende antwoord is: dat weten ze niet. Dat is op Crypto 2004 in Santa Barbara weer eens duidelijk gebleken. Hash-functies die een goede reputatie hadden blijken toch kwetsbaar te zijn. De vraag is dus: waar is de reputatie van een hash-functie op gebaseerd? Dat is vooral op het, ondanks veel onderzoek, ontbreken van een goed idee hoe die hash-functie aan te vallen. Maar zodra iemand zo'n goed idee krijgt, kan een hash-functie plotseling onveilig blijken.
De sterkte van een cryptologische aanval kan gemeten worden in termen van de *werkfactor*, dat is het aantal malen dat een (hash-)bewerking moet worden uitgevoerd om goede kans op een botsing te hebben. Iedere hash-functie heeft in ieder geval last van *brute force*-aanvallen. Daarbij wordt simpelweg van heel veel inputs de hash-waarde bepaald, tot een botsing wordt gevonden. Een hash-functie wordt als cryptografisch sterk beschouwd als er geen aanval bekend is met een kleinere werkfactor dan zo'n brute force-aanval.

Voor het brute force-kraken van de één-richting-eigenschap of de doel-botsing-bestendigheid van een n-bits hash-functie geldt een werkfactor van ongeveer 2n, namelijk het aantal mogelijke hash-waarden.
Botsing-bestendigheid is veel minder moeilijk aan te vallen. Omdat daar twee inputs vrij gekozen kunnen worden, leert de statistiek dat al na het berekenen van ongeveer $1.26 \times 2^{n/2}$ willekeurige hash-waarden een botsing verwacht kan worden. N.B.: $2^{n/2}$ is de vierkantswortel uit 2^n, echt aanzienlijk minder groot dus. Zo'n type brute force aanval heet ook wel een *verjaardags-aanval, of birthday attack*. Die naamgeving komt vanwege het verrassende feit dat in een groep van slechts 23 willekeurige mensen de kans al ongeveer 50% is dat er twee op dezelfde dag jarig zijn. Dat getal 23 is inderdaad ongeveer $1.26 \sqrt{365}$. In het Nederlands wordt *brute force* ook wel *JBF* genoemd: *JanBoerenFluitjes,* oftewel *Just Brute Force.*

Als vuistregel kan gelden dat de bitlengte van een botsing-bestendige hash-functie tweemaal zo groot moet zijn als de sleutellengte van een symmetrisch encryptie-algorithme dat even veilig moet zijn. Als bijvoorbeeld gekozen wordt voor AES-128, dan past daar een hash-functie van bitlengte 256 bij.

Korte geschiedenis van veel gebruikte hash-functies

De eerste in de familie van moderne hash-functies is MD4, door Rivest (de R van RSA) ontworpen in 1990. In 1996 werd aangetoond dat botsingen gevonden kunnen worden met een werkfactor van slechts 2^{20} en daarmee kan MD4 als gekraakt worden beschouwd. Deze hash-functie wordt vrijwel nergens meer gebruikt.
MD5 kan gezien worden als een verbeterde versie van MD4. Van MD5 is al enkele jaren een zwakheid bekend die vooral van theoretisch belang is. Cryptologen raden daarom al enkele jaren het gebruik van MD5 af, ook omdat er goede alternatieven zijn. De Nederlandse cryptoloog Bert den Boer heeft een belangrijk aandeel geleverd in het vinden van deze zwakheden in MD4 en MD5.

Als veilige werkfactor voor kritische toepassingen wordt tegenwoordig 2^{80} beschouwd, hoewel er in standaardiseringskringen al gedacht wordt over 2^{128} (NIST adviseert dit voor ultimo 2010). Dit hangt direct samen met de Wet van Moore, oftewel met het nog steeds in hoog tempo sneller worden van computers. Dat betekent dat voor een kritische toepassing waar botsingbestendigheid een vereiste is, een hash-lengte van 160 bits als minimumvereiste genomen kan worden (over enkele jaren wellicht op te schroeven tot 256 bits). Voor kritische toepassingen waar doel-botsing-bestendigheid voldoet is 128 bits voorlopig nog voldoende. En voor minder kritische toepassingen, waar de veiligheid niet voor een langere periode gegarandeerd hoeft te zijn, kan het misschien wel met minder toe.

Ook de SHA-familie is ontworpen op basis van de ideeën die ten grondslag lagen aan MD4. In de oorspronkelijke variant, SHA-0, werd al snel na publicatie een theoretische zwakheid ontdekt. SHA-0 is daarom nooit in de praktijk gebruikt. Een verbeterde versie is SHA-1, waarop tot voor kort geen serieuze aanvallen bekend waren.

Recente aanvallen op veel gebruikte hash-functies

In 2004 waren er spectaculaire ontwikkelingen te melden op het gebied van hash-botsingen. Volledig onverwacht werden in augustus op de conferentie Crypto 2004 in Santa Barbara van verschillende kanten spectaculaire ontwikkelingen gemeld. Een aantal groepen, onder meer die van Eli Biham en Antoine Joux, liet zien dat gereduceerde versies van SHA-1 aanzienlijk sneller te kraken zijn dan altijd was gedacht. Deze techniek werd gebruikt om botsingen te construeren voor SHA-0.
Ook was er een resultaat van Antoine Joux over het aan elkaar koppelen van verschillende hashfuncties, dat aanzienlijk minder veiligheid geeft dan altijd gedacht. Als klap op de vuurpijl kwam tijdens de 'rump session' (korte, informele, niet vooraf gereviewde aankondigingen) een groep Chinese onderzoekers, onder leiding van Xiaoyun Wang, met concrete voorbeelden van botsingen voor de hash-functies MD4, MD5, en enkele andere. Ze beweerden in minuten tot een uur botsingen te kunnen construeren en dat is een grote doorbraak. Uit de presentatie en het korte artikel dat ze publiceerden werd niet duidelijk hoe ze het precies doen. Uitgebreidere artikelen, waarin ze opening van zaken geven over hun methoden, zullen in mei dit jaar op de conferentie EuroCrypt 2005 in Aarhus (Denemarken) gepresenteerd worden. Hiermee kan ook MD5 als volledig gekraakt beschouwd worden.

Een hash-functie werkt met een groot aantal *rondes*. In elke ronde wordt dezelfde berekening gedaan; de output van een ronde wordt doorgegeven als input voor de volgende ronde. Met een *gereduceerde versie* van een hash-functie wordt bedoeld een versie waarbij een kleiner aantal rondes wordt genomen. Een geslaagde aanval op een gereduceerde versie betekent nog niet dat de volledige versie onveilig is, maar het duidt doorgaans wel op ontwerpzwakheden en is vaak de aanzet voor een aanval op de volledige versie.

Zoals zo vaak zorgt een spectaculair resultaat al gauw voor een golf aan verbeteringen en toepassingen. Het volgende hoogtepunt was al in februari 2005, toen dezelfde groep uit China resultaten aankondigde over de volledige versie van SHA-1. Een gereduceerde versie (tot 58 van de 80 rondes) is gekraakt en er is aangetoond dat de werkfactor om de volledige SHA-1 te kraken niet 2^{80} is, maar 2^{69} (de laatste geruchten spreken zelfs van 2^{66}). Daarmee is het nog niet gelukt om ook daadwerkelijk botsingen voor SHA-1 te maken: de werkfactor is nog flink groot. Maar de toon is gezet en het is te verwachten dat een combinatie van verbeteringen van de aanval en toenemende rekenkracht binnen afzienbare tijd zullen leiden tot daadwerkelijke botsingen voor SHA-1.

De moraal van dit verhaal is: kijk nooit een cryptoloog boos aan als een hash-functie (of, net zo goed, een versleutelingsmethode als Rijndael of RSA) gekraakt of verzwakt wordt. Dat kan altijd gebeuren en is het risico van het vak. Dat alle experts dat risico als laag inschatten neemt niet weg dat de verantwoordelijkheid voor acceptatie van dat risico bij de gebruiker ligt.

Hoe is de situatie van MD5?

De nu bekende resultaten zijn in eerste instantie vooral van belang voor MD5. Botsingen kunnen naar believen gevonden worden met een werkfactor 2^{39} en dat is zo laag dat het binnen een uur rekentijd te doen is.
Hiermee kan wel gezegd worden dat MD5 gebroken is. Maar let op: dat geldt voor botsing-bestendigheid. Voor doel-botsing-bestendigheid ligt het heel anders, daarvoor is de werkfactor nog onveranderd hoog.

Hoe is de situatie van SHA-1?

De nu bekende aanvallen op SHA-1 zijn vooralsnog vooral van theoretisch belang, en hebben op zich geen directe gevolgen voor de veiligheid van toepassingen van SHA-1. Wel is SHA-1 aanzienlijk zwakker dan altijd gedacht: de werkfactor is nu 2^{69} en wellicht zelfs nog maar 2^{66}. Dat is nog steeds behoorlijk groot: als 2^{39} (de werkfactor van MD5 nu) een uur kost, kost 2^{66} zo'n tweehonderdduizend jaar. Wel kan verwacht worden dat de aanval op SHA-1 snel verbeterd zal worden. We zien nu dus het begin van het einde van SHA-1, dat misschien nog niet als gebroken geldt, maar wel als behoorlijk verzwakt.

Kunnen de gevonden hash-botsingen direct kwaad?

In december 2004 zijn twee interessante artikelen verschenen met mogelijke scenario's voor het misbruik maken van hash-botsingen zoals die nu voor MD5 gemaakt kunnen worden. Dan Kaminsky en Ondrej Mikle hebben enigszins vergelijkbare ideeën uitgewerkt, waarbij twee versies van een programma worden aangeboden met dezelfde MD5-hash, maar totaal verschillende functionaliteit. Omdat de MD5-hash hetzelfde is, kan bijvoorbeeld integriteitsbeschermende software als Tripwire effectief om de tuin geleid worden. Men kan van mening verschillen over het realiteitsgehalte van deze scenario's, maar de proof of concepts zijn geleverd.
Arjen Lenstra (Lucent Bell Labs en TU Eindhoven), de genoemde Xiaoyun Wang en ondergetekende hebben in maart 2005 laten zien dat het tamelijk eenvoudig is om publieke RSA-sleutels te maken die een botsing voor MD5 vormen. Deze botsende publieke sleutels kunnen vervolgens worden inge-

bouwd in geldige X.509-certificaten. Op deze manier kan een paar certificaten gemaakt worden met identieke digitale handtekeningen.

Een mogelijk scenario is dat het ene certificaat ter ondertekening wordt aangeboden aan een Certification Authority, die na verificatie van de identiteit van de eigenaar en diens bezit van de bijbehorende privé-sleutel een handtekening zal afgeven. Deze handtekening geldt dan automatisch ook voor het andere certificaat, waar een andere publieke sleutel in staat. De Certification Authority heeft van het tweede certificaat de publieke sleutel niet gezien, en kan niet meer garanderen dat de eigenaar in het bezit is van de bijbehorende privé-sleutel. Een vertrouwende partij die één enkel certificaat krijgt, kan daar op geen enkele manier uit aflezen of het op een dergelijke manier geconstrueerd is of niet en kan er dus niet meer op vertrouwen dat de eigenaar van het certificaat inderdaad in het bezit is van de bijbehorende privé-sleutel.

Hoewel ook hier een realistisch misbruik-scenario nog niet direct bereikt is, is met deze constructie wel een fundamenteel principe van Public Key Infrastructure aangetast.

Welke hash-functies zijn dan nog wel goed?

NIST adviseert SHA-224, SHA-256, SHA-384 en SHA-512 te gaan gebruiken. Die komen uit de SHA-familie en men kan zich afvragen of de nu bekend geworden aanvallen op MD5 en SHA-1 ook werken voor deze hash-functies. Vooralsnog heerst de opvatting dat SHA-224 t/m SHA-512 niet kwetsbaar zijn omdat ze een toch wat andere structuur dan SHA-1 hebben, maar nadere cryptanalytische onderzoekingen zullen daarover de komende jaren duidelijkheid moeten verschaffen. Het lijkt niet onverstandig om voorlopig SHA-256 te zien als de vervanger van SHA-1. Ook standaardiseringsorganisaties als NIST bevelen dat aan. Daarnaast verdient het aanbeveling serieus te gaan kijken naar alternatieven als SHA-512 en Whirlpool.

Advies voor gebruikers van hash-functies

Gebruikers moeten eerst gaan inventariseren in welke applicaties hash-functies gebruikt worden. Vervolgens moet er per applicatie een risicoanalyse gemaakt worden gericht op de toepassing van de hash-functie. Uit deze risicoanalyse moet duidelijk naar voren komen of de applicatie gebruikmaakt van de doel-botsing-bestendigheid of de botsing-bestendigheid van de hash-functie en of aanvallers de inputs zelf kunnen kiezen. Vervolgens moet per applicatie een minimum-werkfactor voor het aanvallen van de hash-functie worden bepaald. Voor minder kritische toepassingen is een werkfactor van 2^{80} nog wel acceptabel, voor kritische toepassingen (bijvoorbeeld digitale handtekeningen waar juridische garanties moeten gelden) kan 2^{128} geadviseerd worden (vergelijkbaar met AES-128-encryptie).
Deze analyse moet door de gebruiker of eigenaar van de systemen gedaan worden en niet door bijvoorbeeld een productleverancier. Het is namelijk geen technisch probleem, maar dit gaat om risico-management op business-niveau, waar de systeem-eigenaar het restrisico zal moeten accepteren. Alleen aan de hand van enerzijds een goede analyse van de toepassing en een inschatting van het risicoprofiel en anderzijds een goede kennis van de stand van zaken in de cryptologie, kan worden bepaald welke hash-functie geschikt is. Ook zal deze beslissing periodiek geëvalueerd moeten worden.
Gebruikers dienen zich ervan bewust te zijn dat iedere hash-functie plotseling verzwakt of gekraakt kan worden en op korte termijn uitgefaseerd en vervangen moet kunnen worden. Daarom is een flexibele en modulaire opbouw van software van groot belang. Het liefst worden meteen al een aantal verschillende hash-functies geïmplementeerd, zodat de gebruiker 'on line' kan kiezen. Zit een bepaalde hash-functie diep ingebakken in de systemen, dan kost het veel om de hash-functie te vervangen en dat zal vaak een reden zijn om het niet te doen. Dat zie je op dit moment in de PKI-wereld, waar bijna alle uitgezette software volledig gebaseerd is op MD5 en SHA-1. Als deze systemen nog lang met de verzwakte hash-functies blijven draaien zal de veiligheid in gevaar kunnen komen.
Trucs zoals het combineren van verschillende hash-functies zijn in beginsel af te raden. Het komt regelmatig voor dat dat soort constructies niet de veiligheid bieden die op het eerste gezicht gesuggereerd wordt, zoals Joux op Crypto 2004 aantoonde. Het is beter om gestandaardiseerde protocollen te gebruiken; daarvan mag redelijkerwijs verwacht worden dat ze goed zijn geanalyseerd en als ze gekraakt worden, wordt dat meteen breed bekendgemaakt.
Gebruikers van hash-functies schaffen vaak standaard-software aan en maken zich daarmee zeer afhankelijk van hun systeem- en product-leveranciers. Gebruikers moeten daarom hun leveranciers voortdurend lastig blijven vallen met vragen naar onder meer de flexibiliteit van hun oplossing: is het systeem makkelijk aan te passen voor een andere hash-functie met een andere lengte?

Advies voor gebruikers en aanbieders van certificaten

MD5 had al niet meer gebruikt moeten worden vanwege twee argumenten: MD5 heeft geen werkfactor van 2^{80} maar slechts van 2^{64} en een ontwerpzwakheid was al bekend. Sinds Crypto 2004 komt daar een dodelijk derde argument bij: er zijn botsingen te maken in overvloed.
Aangetoond is dat deze hash-botsingen direct en eenvoudig zijn te gebruiken om een fundamenteel principe van PKI aan te tasten. Aan een enkel certificaat is niet te zien of het met behulp van een MD5-botsing gemaakt is of niet. Om al deze redenen is het onverantwoord om nu nog MD5 te gebruiken als hash-functie bij het aanmaken van digitale certificaten. Dat geldt zeker bij kritische toepassingen, bijvoorbeeld waar juridische onloochenbaarheid van belang is.
Aan de andere kant is de doel-botsing-bestendigheid van MD5 nog redelijk overeind gebleven. Dat betekent dat een MD5-hash-waarde waarvan in alle redelijkheid kan worden aangenomen dat die niet met het oog op een botsing gemaakt is, nog wel als veilig gezien kan worden. Een goed advies lijkt nu te zijn om een datum vast te leggen, zeker voor Crypto 2004 (bijvoorbeeld 1 augustus 2004) en alle gebruik van MD5 na die datum als mogelijk onveilig te beschouwen, want mogelijk een geconstrueerde botsing.

Alle gebruik van MD5 voor die datum kan als veilig beschouwd worden, want heeft hooguit last van een doel-botsing-aanval en die heeft nog steeds een voldoende hoge werkfactor.
Dat betekent dat certificaten op basis van MD5 uitgegeven voor 1 augustus 2004 nergens last van hebben en gewoon vertrouwd kunnen worden tot het einde van hun geldigheid. Maar certificaten op basis van MD5 uitgegeven na 1 augustus 2004 zouden niet meer vertrouwd moeten worden door eindgebruikers. De uitgevende Certification Authorities doen er verstandig aan deze certificaten in te trekken. Certification Authorities moeten zeker geen MD5 meer gebruiken voor het aanmaken van hun certificaten. De Certification Authority kan wel zeggen dat de certificaten niet geconstrueerd zijn voor een hash-botsing, maar de gebruikers hebben geen enkele mogelijkheid dat te controleren.
Zolang de uitfasering van SHA-1 nog niet voltooid is, kan het beste per direct overgestapt worden van MD5 op SHA-1. Daarnaast verdient het aanbeveling werk te gaan maken van de vervanging van SHA-1 door bijvoorbeeld SHA-256. Daarvoor is nog even tijd, maar niemand weet hoe lang. NIST geeft 2010 aan als horizon voor SHA-1. Aangezien inmiddels SHA-1 verder verzwakt is, zou ik nu hooguit eind 2006 adviseren voor het volledig uitgefaseerd hebben van SHA-1.

Advies voor leveranciers van cryptografische producten

Een leverancier van cryptografische software die nu nog producten met MD5 erin voor veilig verkoopt, misleidt zijn klanten. Een leverancier van hashing software die niet druk bezig is (geweest) met het implementeren van alternatieven voor SHA-1, verdient ook met wantrouwen benaderd te worden. Cryptografische software-producenten doen er goed aan de architectuur van hun producten onder de loep te nemen, en aan te passen aan de benodigde flexibiliteit voor het snel kunnen wisselen van hash-functies. Ze dienen een voldoende ruim aanbod aan hash-functies te hebben en de stand van zaken in de academische cryptologie en de standaardiseringswereld voortdurend te monitoren.

Conclusie

MD5 is dood. Jaren geleden, toen theoretische zwakheden in MD5 werden blootgelegd, begonnen de cryptologen al te roepen dat het gebruik van MD5 afgeraden diende te worden. Nu er daadwerkelijk botsingen in grote hoeveelheden gemaakt kunnen worden, en er al enkele voorzichtige aanvalsscenario's verschijnen, zou niemand MD5 nog actief moeten gebruiken. Waar MD5 nog stuiptrekkingen vertoont dient het de genadeslag te krijgen.
SHA-1 is stervende. Het is nog niet duidelijk hoe lang het stervensproces gaat duren. Het is nog niet per se risicovol om nu nog SHA-1 in te zetten, maar er dient wel vanaf nu serieus gewerkt te worden aan de vervanging van SHA-1. Dat zal, zowel voor gebruikers als voor cryptografie-leveranciers, een hoop extra werk betekenen. Het is te hopen dat lessen geleerd worden en systemen flexibeler, minder afhankelijk gemaakt worden van een slechts beperkte keuze in cryptografische primitieven zoals hash-functies.
SHA-256, SHA-512 en Whirlpool lijken voorlopig de erfgenamen. Het is te verwachten dat de aandacht van cryptanalytici zich de komende tijd op deze hash-functies zal gaan richten.

URLs

- Het oorspronkelijke artikel van de Chinese onderzoeksgroep zoals dat op 17 augustus 2004 bekend werd gemaakt, is te vinden op *http://eprint.iacr.org/2004/199.*
- Een korte aankondiging van de aanval op SHA-1 uit februari 2005 staat op *http://islab.oregonstate.edu/koc/ece478/x/shanote.pdf.*
- Het genoemde artikel van Dan Kaminsky is te vinden op *http://www.doxpara.com/md5_someday.pdf.*
- Het genoemde artikel van Ondrej Mikle is beschikbaar op *http://eprint.iacr.org/2004/356.*
- Een recent artikel van Arjen Lenstra met praktisch advies over het gebruik van hash-functies staat op *http://cm.bell-labs.com/who/akl/hash.pdf*
- De botsende X.509-certificaten van Lenstra, Wang en De Weger zijn met het nodige achtergrondmateriaal te vinden op *http://www.win.tue.nl/~bdeweger/CollidingCertificates.*

Nieuws

Nieuwe mbo-opleiding digitaal rechercheren

In augustus dit jaar starten verschillende regionale opleidingscentra (ROC's) met een nieuwe opleiding digitaal rechercheren. De digitaal rechercheur is de Sherlock Holmes op ict-terrein. Hij of zij signaleert en onderzoekt cyber-criminaliteit. Dit door nieuwe onderzoeken te initiëren, digitaal bewijsmateriaal veilig te stellen en onderzoek te verrichten naar en op bewijsmateriaal. Geen overbodige luxe met de snelle toename van virussen, samenspanning en andere vormen van digitale fraude en criminaliteit. De mbo-4 opleiding duurt drie tot vier jaar, afhankelijk van het studietempo van de leerling. De opleiding zal in Nijmegen, Eindhoven, Zeeland en Utrecht worden aangeboden. Februari vorig jaar heeft ECABO, het landelijk kenniscentrum voor het middelbaar beroepsonderwijs, het beroepscompetentieprofiel (bcp) behorend bij de opleiding digitaal rechercheren van de ROC's goedgekeurd. De vertaling van het bcp naar het kwalificatieprofiel is nu in volle gang. "We overleggen momenteel intensief met de politie. De politiekorpsen leiden ook digitale rechercheurs op. Daarom moeten we goed met elkaar afspreken wie wat voor z'n rekening neemt. Een digitale politierechercheur doet vooral tactisch onderzoek gericht op strafzaken. 'Onze' digitale rechercheurs werken vaak binnen bedrijven. Als blijkt dat er een strafzaak nodig is, dan dragen wij het over aan de politie", aldus Hans Blankendaal, senior adviseur ict bij ECABO.
- bron - Security.nl