Stelsel van certificaten is eigenlijk een gedistribueerde database

Database door ondertekening

Rick van Rein

In DB/M 6 bespraken we DNS als (bizarre) vorm van een database. Ditmaal gaan we in op een soortgelijk onverwachte vorm, namelijk digitaal ondertekende statements.

Overal op de wereld kom je data tegen, al dan niet gestructureerd, en daar kunnen links tussen worden gelegd. Steeds vaker wordt XML als database gezien; een XML-document heeft een interne structuur die door computers te interpreteren valt, en daarnaast kunnen er door middel van URL's koppelingen worden gelegd naar andere documenten. Wie bereid is dat als database te beschouwen kan ook met de hier getrokken parallel uit de voeten. Digitaal ondertekende data bestaan doorgaans uit een stuk data, een vermelding van een handtekeningalgoritme en een daarmee geconstrueerde digitale handtekening. Die digitale handtekening is een code die alleen door de ondertekenaar kan worden gemaakt, en die gekoppeld is aan de ondertekende data – de handtekening is dus niet voor andere data bruikbaar, zoals met enig creatief knip- en plakwerk wel mogelijk zou zijn met ondertekende faxberichten.

De structuur van ASN.1

Het idee van XML is dat het leesbare documenten levert die bovendien door de computer te interpreteren zijn. Wie een computergegenereerd document opent leert echter al snel dat de leesbaarheid ver te zoeken is – een HTML e-mail is een uitstekend voorbeeld, zoals liederlijk uiteengezet op http://dustman.net/andy/HTMLMail – dus eigenlijk is een XML-document ofwel computergegenereerd, ofwel leesbaar. XML is ontworpen voor de uitwisseling van data in een standaardformaat. ASN.1 is een oudere standaard voor de codering van data, met datzelfde doel van uitwisselbaarheid. Deze standaard heeft niet tot doel om leesbaar te zijn voor mensen (hoewel de hexdumps best leesbaar zijn) maar die wel probeert om data af te beelden. Gegeven het feit dat de helft van de XML-documenten toch niet leesbaar is voor mensen, is ASN.1 op diverse fronten een betere datarepresentatie dan XML.
De algemene structuur van een ASN.1 in binaire vorm is TLV – tag, length, value. Dat houdt in dat de soort data in een tag wordt gecodeerd, gevolgd door een lengte en dan evenveel bytes aan value als aangegeven in de lengte. Dat voorkomt een hoop problemen die XML ondervindt met quotes en de tekens voor 'kleiner dan' en 'groter dan'. Dergelijke problemen zijn natuurlijk op te lossen met escapes (&quote; en < en >) waarbij natuurlijk de escape ook 'geëscaped' moet worden (& voor een ampersand). Maar dan nog is het soms lastig om te bepalen op welke plek een code wordt omgezet. Als een XSLT-mapping nodig is om de data om te zetten in HTML, dan moet een 'kleiner dan' teken in HTML bijvoorbeeld weergegeven worden als " zodat er na XSLT " overblijft. Het probleem hieraan is dat de codering in XML afhankelijk wordt van het aantal bewerkingsstappen, en normaliter wil je dat niet vastpinnen op het moment dat je je data noteert. Escapes leiden dus tot onnodige problemen, en een TLV-codering voorkomt dergelijke sores.

Er zijn diverse voorgedefinieerde tags in ASN.1, voor standaardzaken als sequenties en verzamelingen. Ook hier is ASN.1 superieur aan XML, waarin alles sequentieel wordt voorgesteld. De hieruit volgende noodzaak tot overspecificatie in XML betekent dat bijvoorbeeld een XML opslaande engine verplicht wordt om informatie over de volgorde van tags op te slaan. Dit levert een bekend probleem bij de opslag van XML in een RDBMS: De afgedwongen volgorde vertraagt query's zonder dat dit in alle gevallen nodig is.
De wijze waarop ASN.1 sequentiële data noteert, is door in het value-deel van de sequentie weer een reeks TLV-waarden volgens ASN.1 te noteren. Op deze manier zijn dus geneste datastructuren te noteren, net als met XML. Anders dan bij XML ligt de binaire representatie niet vast; de genoemde TLV-waarden zijn eigenlijk een bepaalde representatie van ASN.1, en wel de veelgebruikte DER-representatie. DER staat voor Distinguished Encoding Rules. Zo'n notatie dient bijvoorbeeld om overdraagbare data weer te geven.

Gebruikerstypes in ASN.1

Als data in ASN.1 worden gerepresenteerd, dan zit men niet vast aan een beperkt aantal voorgedefinieerde tags. Er is een mogelijkheid om zelf types te definiëren. Dat gebeurt door middel van Object Identifiers, of kortweg OID's. Dit zijn waarden uit een numerieke boom. Alle OID's die beginnen met 1.3.6.1.4.1.10471 zijn bijvoorbeeld van OpenFortress, met 1.3.6.1.4.1.10471.6 voor security-gerelateerde tags en daaronder weer 1.3.6.1.4.1.10471.6.2.3 voor een intern gedefinieerde referentie naar een PGP-sleutel. Dergelijke gebruikerstypes kunnen worden opgenomen in een ASN.1-specificatie; bijvoorbeeld kan een veld worden ingeleid

door een dergelijke specificatie van de erin bevatte waarde. Doordat een hele tak van de OID-boom wordt toegewezen aan een organisatie, blijven de OID-waarden wereldwijd uniek, zelfs al maakt elke organisatie eigen types aan. Er is een vertaling van namen (zoals pgpKeyReference) naar OID-codes (zoals 1.3.6.1.4.1.10471.6.2.3) mogelijk, maar in tegenstelling tot de OID-codes zijn de gebruikte namen dan wel lokaal. Ze volstaan echter voor menselijke interactie met ASN.1.
Hoewel het mogelijk is om volledig eigen ASN.1 types te gebruiken en die door eigen software te laten verwerken, is het met ASN.1 ook prima mogelijk om standaardtoepassingen te gebruiken. Voor dergelijke toepassingen zijn ook standaard OID-codes met namen gedefinieerd en gestandaardiseerd; voorbeelden zijn commonName, organization, country en emailAddress.
Een record in ASN.1 is gewoon een sequentie van OID-gecodeerde waarden:

```
SEQUENCE
  commonName=Orvelte's Warme Bakker
  locality=Orvelte
  country=NL
  pgpKeyReference=http://orvelte.nep/bakker
  Encrypt uw online bestellingen met deze sleutel
```

Maar het kan bijvoorbeeld ook een verzameling van alternatieve waarden zijn:

```
SET
  SEQUENCE
   commonName=Orvelte's Warme Bakker
   locality=Orvelte
   country=NL
  SEQUENCE
   emailAddress=bakker@orvelte.nep
```

Net als bij XML, is het ook met ASN.1 mogelijk om software te ontwerpen om delen over te slaan die niet herkend worden. In de TLV-notatie is het dankzij de lengte mogelijk om de onbekende data over te slaan en door te gaan met de daaropvolgende data. Of dit wenselijk gedrag is kan per toepassing worden beslist.

Certificaten

De meest bekende toepassing van ASN.1 is het certificaat; dat wordt gebruikt om TLS-servers (zoals secure websites) te beschermen. Die bescherming bestaat uit de codering van het verkeer tussen client en server, maar zeker ook uit de authenticatie van de server. Pas als vaststaat dat de server werkelijk die van de Postbank is kan er immers op worden vertrouwd dat gecodeerde data alleen door de gewenste partij te decoderen zijn.
Een certificaat is de eerder genoemde combinatie van drie waarden: ondertekende data, een ondertekeningsalgoritme en een handtekening. Het ondertekeningsalgoritme is een standaard OID, de handtekening is een BLOB (dat heet in ASN.1 een BIT STRING) en de te ondertekenen data bevatten gegevens over de ondertekende identiteit, maar ook die van de ondertekenaar. Plus de nodige administratieve details.
Een ondertekende identiteit kan bijvoorbeeld de sequentie van datavelden zijn die hierboven gegeven was als voorbeeld van de warme bakker in Orvelte; daarmee is dus een record met data in een certificaat opgenomen. De ondertekenende partij wordt met een soortgelijke beschrijving vermeld in het certificaat:

```
SEQUENCE
  commanName=Orvelte's Veldwachter
  locality=Orvelte
  country=NL
  emailAddress=bromsnor@orvelte.nep
```

Ook deze veldwachter heeft weer een eigen certificaat dat hem identificeert. Bij dat certificaat hoort ook een sleutel, waarmee hij de handtekening construeert die het certificaat van de warme bakker completeert. De vermelding van deze informatie in het certificaat van de bakker is dus eigenlijk een referentie naar de sleutel van de veldwachter.

> Het stelsel van certificaten dat wordt gebruikt voor secure webservers is eigenlijk een gedistribueerde database

Het certificaat van de veldwachter moet op zijn beurt weer ondertekend worden, maar dat zou eeuwig door kunnen gaan. Ergens eindigt het met een certificaat dat voor zichzelf ondertekent, en dat dan als zogenaamd root-certificaat kan worden gebruikt.

Hiërarchische database

Hoewel primitief, vormen certificaten toch een eenvoudige database, want er is sprake van records met onderlinge (hiërarchische) links. Ter optimalisatie van de links hoort bij de administratieve extra's ook vaak een numerieke terugverwijzing naar de partij die ondertekent.
Dit systeem heeft een heel bijzondere eigenschap. Er is namelijk geen centrale opslag nodig om zeker te weten dat de data kloppen. Zodra een certificaat uitgegeven is kan het worden gedistribueerd, en toonder kan ermee aantonen dat de geclaimde data in het ondertekende record kloppen. Daarbij is het van vitaal belang dat de digitale handtekening bij de ondertekende data hoort – het is hierdoor niet mogelijk om de data te veranderen zonder daarbij de handtekening ongeldig te maken. De handtekening valideert dus de herkomst van het certificaat, zonder dat die op de plek van herkomst nagevraagd hoeft te worden. De relaties tussen certificaten zijn vertrouwensrelaties. De warme bakker claimt zijn identiteit en verwijst ter controle van die claim naar de veldwach-

ter als betrouwbare instantie. Dankzij de handtekening is die link valideerbaar voor iedereen die de veldwachter vertrouwt, zonder daarbij de veldwachter lastig te hoeven vallen.
Het stelsel van certificaten dat wordt gebruikt voor (met name) secure webservers is eigenlijk een gedistribueerde database. De kennis in die database behelst het vertrouwen in het eigendom van een website – ofwel, dat we werkelijk met de Postbank communiceren. De selectie van root-certificaten (zoals dat van de veldwachter) bepaalt welke subset van de mogelijke veilige sites we impliciet vertrouwen, dat wil zeggen zonder waarschuwende pop-up. Voor het gemak (maar beslist niet voor de veiligheid) worden bij browsers of operating systemen standaard al wat van die root-certificaten geïnstalleerd.

PGP slimmer en flexibeler

Een veel verfijndere vertrouwensdatabase is het Web of Trust dat door PGP-gebruikers in stand wordt gehouden. Zoals het bij certificaten mogelijk is slechts één ondertekenaar te erkennen, zo is het met PGP mogelijk om een willekeurig aantal ondertekenaars te hebben op één en dezelfde sleutel.
PGP werkt niet met ASN.1 als notatie, maar het gebruikt een eigen representatie die daar wel op lijkt. Die notatie is bedoeld om zo flexibel mogelijk met ondertekende statements uit de voeten te kunnen. Zo is het heel eenvoudig om e-mail te ondertekenen alvorens het te versturen. Zo'n e-mail verwijst dan naar een sleutel, die op zich naar een hele rits ondertekenaars verwijst, die op zich weer naar andere sleutels verwijzen.

> Bij alle systemen is het mogelijk om door middel van een 'signing policy' expliciet een doel voor de ondertekening vast te leggen

De ondertekeningsrelaties van PGP zijn ook voornamelijk vertrouwensrelaties, maar doordat er alternatieven zijn is er meer vrijheid voor de ontvanger (van bijvoorbeeld een PGP-ondertekende e-mail) om de herkomst te valideren. Het is bijvoorbeeld mogelijk om meerdere paden parallel te eisen tussen de sleutel van de afzender en die van vertrouwde instanties, inclusief jezelf. Er is dus meer interpretatie mogelijk over de betekenis van de relaties. Daarnaast is het met PGP mogelijk om meer soorten handtekening te zetten dan alleen voor identiteitscontrole. Er zijn bijvoorbeeld time stamps (dus bekrachtigde tijdwaarnemingen) en handtekeningen op handtekeningen. Bij alle systemen is het daarnaast mogelijk om door middel van een 'signing policy' expliciet een doel voor de ondertekening vast te leggen. En het algemene principe van digitale handtekeningen ligt niet helemaal vast op vertrouwen of op identiteit. Het is gewoon de meest voor de hand liggende relatievorm tussen onafhankelijk beheerde domeinen. Een handtekening is natuurlijk wel altijd bedoeld ter bekrachtiging, meestal over de grenzen van zulke domeinen heen.

Alleen toevoegen

Het is niet mogelijk om een ondertekende verklaring terug te draaien. Dat volgt uit de gedistribueerde aard van de informatie: louter afwezigheid van informatie wil niet zeggen dat er nooit iemand zal proberen om een oude link nieuw leven in te blazen. In principe kun je in een stelsel van ondertekende statements dus alleen maar kennis toevoegen.

Natuurlijk is er wel een mouw aan te passen als er ook verwijderd moet kunnen worden. Wat dan gebeurt is dat er een verwijdering wordt geplubliceerd, in toevoeging op de eerdere ondertekening. Als die naast elkaar staan dan kunnen ze tegen elkaar worden weggestreept. Het is dan wel zaak te zorgen voor de wereldwijde vindbaarheid van de verwijdering, en daarin schuilt vaak een probleem. Eigenlijk wil je ondertekende statements maken op informatie die altijd geldt; een manier om dat te doen is een verloopdatum te stellen voor de ondertekening. Bij certificaten is dat standaard, bij PGP is het een mogelijkheid.

DNSsec

In DB/M 6 schreven we over DNS als een gedistribueerde database. Ook hier wordt gewerkt aan ondertekende informatie, vooral omdat het zeer simpel is voor kwaadwillenden om DNS over te nemen. Wie een DNS-verzoek als eerste beantwoordt wordt doorgaans geloofd. Met DNSsec, ook wel Secure DNS genoemd, is het mogelijk een handtekening toe te voegen aan een DNS-antwoord. Zo'n handtekening wordt gezet door een bovengelegen *signing authority*, zodat vaststaat dat de informatie van de juiste plek afkomstig is. Wie deze handtekeningen controleert kan er dus op rekenen dat de verkregen informatie van goede komaf is. De match tussen de gedistribueerde DNS-database en de gedistribueerde vertrouwensdatabase door ondertekening is heel natuurlijk. De systemen zijn een goed koppel doordat ze allebei hun verantwoordelijkheden kunnen distribueren naar andere domeinen, die mogelijk onder andermans beheer vallen. Toch is met de introductie veel infrastructurele omslag gemoeid, en dat houdt de introductie al jarenlang tegen.

Tot slot

Digitale handtekeningen kunnen worden gezien als een link tussen een ondertekend record en een ondertekenaar die de verantwoordelijkheid neemt over dat record. Nadat een handtekening geplaatst is kan die kennis de wereld in gezonden worden, en bijdragen aan een virtuele database die is opgebouwd uit losse statements.
Het geheel combineert uitstekend met DNS, waarin ondertekeningen kunnen voorkomen dat de resolutie van bijvoorbeeld domeinnamen naar IP-adressen en mailservers wordt *gespoofd*. Ondertekening slaat op gedistribueerde links die meestal gaan over vertrouwen in de ondertekende data. Toch is het mogelijk om ook andere zaken in een ondertekend statement op te nemen.

Rick van Rein
Dr. ir. H. van Rein (rick@openfortress.nl) is ontwikkelaar en beheerder bij OpenFortress Digital signatures.

Update

Netezza en IBM leveren verbeterde gegevensintegratiefunctie

Ondersteuning door Netezza's high-speed dataloader maakt het gebruikers van IBM Information Server mogelijk om de prestatie van het NPS-systeem volledig door te voeren in hun gegevensintegratie. Door deze integratie kunnen klanten IBM Information Server naadloos doorvoeren in het Netezza datawarehouse-platform en tegelijkertijd de prestatie maximaliseren. IBM Information Server ondersteunt Netezza's high-speed data loader, die momenteel een gegevensoverdrachtssnelheid van 500 Gigabyte per uur bereikt, en stelt IBM Information Server-klanten in staat een zeer hoge verwerkingscapaciteit te bereiken bij het laden van gegevens in het NPS-systeem.
Het Performance Server-systeem van Netezza is een datawarehouse-platform dat speciaal ontwikkeld is voor sneller en goedkoper analyseren van Terabytes aan gedetailleerde gegevens dan bestaande datawarehouse-opties. Het NPS-systeem slaat Terabytes aan dossiers op en filtert en verwerkt deze in één eenheid, waarbij alleen de relevante informatie voor elke vraag wordt geanalyseerd. Netezza heeft CPU-kracht naast de gegevens geplaatst, waardoor het NPS-systeem razendsnel processen uitvoert waarmee de meeste datawarehouse-systemen uren of zelfs dagen mee bezig zouden zijn en waardoor een spectaculaire toename van productiviteit binnen de organisatie mogelijk is.

Clientele ITSM integreert BI-toepassing

Mproof, ontwikkelaar en leverancier van Clientele software voor onder meer IT service management, klantsupport en self service, biedt BI-toepassingen als nieuwe uitbreiding binnen Clientele ITSM. Deze zijn gebaseerd op standaard Microsoft SQL 2005-technologie. Voor de nieuwe functionaliteit zijn geen kostenintensieve tools van derde partijen nodig. Ter verbetering en uitbreiding van de huidige Crystal Reports rapportagemogelijkheden in Clientele is een datawarehouse gebouwd, inclusief OLAP cubes, dat data ontsluit uit Clientele ITSM. Hierdoor kunnen gebruikers op een laagdrempelige manier ad hoc analyses en rapportages uitvoeren. Dit zijn bijvoorbeeld trendanalyses met betrekking tot support calls en performance metingen aan de hand van KPI's (Key Performance Indicators) ten behoeve van het service level management. Het datawarehouse, de OLAP cubes en het ETL-proces zijn ontwikkeld op basis van het Microsoft SQL Server 2005 platform. Als front-end tool kan gebruik worden gemaakt van Microsoft Excel, maar ook compatibiliteit met diverse andere front-end tools is standaard.
Zie www.clientele-itsm.nl