Enterprise

**Tweeënhalf jaar geleden kondigde Sun aan dat Java Systems Acces Manager open source zou worden. Nu is dat al enige tijd praktijk. Versie 8.0 van de commerciële versie zal zelfs dezelfde code bevatten als de open source-versie. Pat Patterson, federation architect bij Sun Microsystems, vertelt er meer over.**

# Hoe open is Open SSO?

## Interview met Pat Patterson

**Dré de Man**

*Single sign on, oftewel SSO is een goed idee, maar toch zie je het minder dan je zou verwachten.*

Ik zou dat zo niet durven te zeggen, binnen een enterprise context zien we veel deployments, en zelfs daarbuiten. Het punt is dat wanneer het goed gedaan is, je niet eens merkt dat het er is. Je logt in en je haal je resources op maar de gebruikers is er zich niet van bewust dat hij van de een naar de andere website gaat. Uiteraard komt het nog voor dat je apart moet inloggen voor je salarisapplicatie, dan voor je portal, dan voor een andere webapplicatie. Maar single sign on zorgt voor veel gebruiksgemak en wint daardoor aan populariteit.

*Wordt single sign on niet te gecompliceerd wanneer je het over diverse services wil gebruiken, dus in een SOA?*

De klassieke single sign on vind je binnen één organisatie. Met federatie kunnen je dat uitbreiden tot andere ondernemingen, en leveranciers of iets dergelijks. In sommige use cases werkt het heel goed, bijvoorbeeld bij outsourcing. Daar wil je gebruikers log in en toegang geven alsof het een deel van je bedrijf zou zijn. Wamt ook al heb je HR-functie geoutsourced, dan wil je daar toch nog gewoon vanaf je bedrijf toegang toe hebben. Bij meer consument-achtige federatie moet je wat voorzichtiger zijn. Wanneer je inlogt bij je bank en dan resources bij je aandelenmakelaar gebruikt, dat roept wat vragen op in de zin van dat je nu gebruikers traint om in te loggen op de één plaats en resources te gebruiken op de andere. Wanneer je niet voorzichtig bent, dat kan het heel erg lijken op phishing, want in feite doe je dat. Een andere trend die we nu zien is het vervangen van gebruikersnaam en password door authentificatiemogelijkheden die beter bestand zijn tegen phishing.

*Zoals?*

Hardware tokens. Er bestaan systemen waar je je username intikt en dan een sms krijgt met de respons die je daarna weer moet intikken, dat soort technieken. Een password is nu eenmaal geen echt sterk bewijs dat je degene bent voor wie je je uitgeeft. Onafhankelijk van de complexiteit van een password, wanneer je het kunt oppikken, dat ben je er. Dat is een principieel verschil tussen een password en een channel reponse systeem. Wanneer je in zo'n systeem precies afvangt wat er gebeurt, dan heb je daar niets aan voor de volgende keer dat je toegang wil. Single sign on is heel populair bij deploymenst bij de overheid, zoals het burgerportaal in Noorwegen, Mon Service Public in

Frankrijk en in België. Het lijkt heel goed te werken in die situaties waarin je een aantal silosystemen hebt, allemaal deel uitmakend van de overheid. Het voordeel van één sign on is ook dat de password policy strenger kunt maken, want je maakt het minder waarschijnlijk dat een gebruiker het password gaat opschrijven.

*Hoe eenvoudig is het wanneer een ontwikkelaar er gebruik van wil maken?*
We werken er hard aan om het met iedere release simpeler te maken. Eén van de dingen die we in de voorlaatste release hebben gedaan, is ervoor te zorgen dat het als een enkel war document gedeployed kan worden. In plaats van een installer te gebruiken, plaats je de war file in je Java container, van Tomcat of zo. Dan ga je naar de URL en vult wat basis configuraties in en het draait. Het werkt zo dat je een centrale Open SSO-server hebt, waar je policies bijhoudt. Dan heb je agents die je in de containers deployt waar je applicaties draaien, in Apache, of Glashfish, of waar dan ook. Als dat draait en je Open SSO op je gebruikers repository hebt gericht en alle gebruikers toegang tot de server hebt gegeven dan ben je er in grote lijnen.

## Cookie

De gebruiker wil toegang tot de resource die je net beschermd hebt, de agent werkt als een filter, het eerste wat het doet is kijken of de cookie in de request aanwezig is, zo niet dan stuurt het je naar de authenticator, de centrale Open SSO. Open SSO stuurt je terug, maar nu met een cookie, nu gaat het naar de server en vraagt: mag de user die door deze cookie gepresenteerd wordt toegang tot die resource hebben? De server kijkt naar de policy en zegt vervolgens ja of nee. Belangrijk: het cookie heeft geen echte inhoud, er is alleen een toevallig gegenereerd getal dat we als *handle* gebruiken naar de session tabel die centraal bijgehouden word. Die tabel heeft alle informatie over wie je bent, wanneer je ingelogd hebt en hoe, het kan die informatie ook terug sturen naar de agent om het in de http headers te inserten. Webapplicaties weten dat ze beschermd zijn want als het request zo ver is gekomen moet iemand geauthentificeerd zijn, en zien ze in de http-headers wie de geauthenticeerde gebruiker is. Het maakt het voor ontwikkelaars heel gemakkelijk om hun applicatie te beschermen en authorisatie en authentificatie toe te passen. Wat we ook doen: met de Java EE-container pluggen we meteen in op het Java security systeem. Het zet alle rollen en zo op, dus je kunt declaratieve security doen in Java EE. De ontwikkelaar hoeft zich er dus nauwelijks mee bezig te houden en kan zich in zijn applicatie echt richten op het businessprobleem dat hij moet oplossen.

## Webservices

*Voor benaderen van webservices ziet Patterson drie patronen:*
Net zoals je webapplicaties kunt benaderen, kun je ook webservices benaderen. Er zijn daarvoor drie patronen. Bij de eerste heb je een webservice consumer en een web service provider en je wil requests doen. Het eerste *pattern* is: doe het allemaal voor me. Ik wil me er niet mee bezig houden, het moet een configuratie zijn, een laag boven op de dingen die ik doe. Het systeem zorgt ervoor dat mijn uitgaande request vergezeld wordt van een SAML-assertion, terwijl deze bij binnenkomende assertions gecontroleerd moeten worden.
Het tweede pattern is het soort standaardgebaseerde patroon waarmee ik identity based services wil aanbieden. Dat is waar Liberty ID-WSF gebuikt wordt. Je wil een discovery service, kijken waar een agenda van een bepaalde gebruiker is, en zijn persoonlijke profel. Je wil services implementeren die afhankelijk zijn van identiteiten en geo-locatie. Je wil een service die voor een bepaalde user de locatie teruggeeft, maar met alle garanties voor privacy, permissies et cetera. Dat is een model waarbij je de tamelijk veelomvattende flexibele identiteitsstandaarden zoals ID-WSF wil gebruiken. Dan is er een derde model dat we in het project nu gebruiken, om het voor ontwikkelaars gemakkelijk te maken om identiteiten te manipuleren. Soms wil je in je applicatie een gebruiker authentificeren en vanuit de programmatuur een authentificatie webservice aanroepen en andere dingen doen. In feite wil je zeggen: is de gebruiker die door dit token wordt gerepresenteerd toegestaan om dit of dat document te lezen of weg te schrijven et cetera. We implementeren dit als SOAP en REST webservices om ontwikkelaars in welke programmeertaal dan ook toegang te geven tot deze basis functies, om het gemakkelijk te maken om applicaties te schrijven die in feite werken met identiteit.

*U zit vaak tussen twee belangen in: privacy aan de ene kant en security aan de andere. Is dat niet een moeilijke positie?*
Ik denk het niet. Veel van het werk dat bij de Liberty Alliance wordt gedaan, richt zich specifiek op deze twee dingen. Als je kijkt naar de manier waarop SAML werkt, het security aspect is dat je een vertrouwensrelatie hebt tussen de service provider die toegang tot een resource aanbiedt en de identiteitsprovider die gebruikers authentificeert. Het is een klassieke security, net als Kerberos, waar je die functionaliteit delegeert. Maar in de situatie waarin je een identeitsprovider hebt en meerdere service providers kunnen meerdere onwenselijke dingen gebeuren. Een ervan is dat iemand binnen je *circle of trust*, het doel van de authentificatie niet wil dat een gebruiker een globale unieke identifier heeft. Want dan kunnen de diverse providers dingen delen en een dossier gaan samenstellen over een gebruiker. Dus wanneer je die link tussen twee accounts creëert dan genereert het gewoon een heel lang random getal, en iedere kant moet dus met je account in verband brengen. Binnen de context van die relatie geldt dat getal, wanneer naar een andere service provider gaat is het een ander lang getal. De indentifier moet die bijhouden en de juiste doorgeven bij een single sign on. Dit is allemaal ingebouwd. Je kunt zelfs transient relaties hebben waarbij de service provider niet eens verband kan leggen tussen verschillede bezoeken van dezelfde gebruiker. De combinatie van privacy en security wordt dus steeds belangrijker en er zijn steeds mooiere oplossingen voor. «

# Patches Patches Patches Patches Patches Patches Patches P

Artikelen over onderwerpen als software-ontwikkeling, Java, UML, eXteme Programming en nog veel meer vindt u in het Online Archief van Array Publications. Vaktijdschriften als Software Release, Java Magazine, Database Magazine en ons Oracle vakblad Optimize hebben hun artikelenarchief online gezet. Dankzij de heldere zoekstructuur vindt u snel wat u zoekt op www.release.nl.

## Sun Microsystems neemt virtualisatie-aanbieder innotek over

Sun Microsystems neemt het Duitse innotek, aanbieder van open source virtualisatiesoftware, over op basis van aandelenaankoop. Met de software van inotek,VirtualBox genaamd, wordt Sun's xVM-platform verder uitgebreid naar de desktop, waardoor ontwikkelaars efficiënter applicaties kunnen bouwen, testen en gebruiken op meerdere platforms. Hiermee versterkt Sun zijn positie in de virtualisatiemarkt. innotek's opensource-product VirtualBox heeft zich, met meer dan vier miljoen downloads sinds januari 2007, gevestigd als een van de vooraanstaande ontwikkelplatforms voor desktopvirtualisatie. Het valt vooral op door zijn compactheid: 20 MB. Met VirtualBox is het mogelijk op desktops of laptops meerdere besturingssystemen, zoals Windows, Linux, Mac of Solaris, naast elkaar te gebruiken. Met slechts één muisklik kan worden gewisseld tussen de besturingssystemen Softwareontwikkelaars kunnen hiermee eenvoudiger uitgebreide applicaties voor verschillende platforms bouwen en testen. "VirtualBox is voor Sun de perfecte aanvulling op ons recentelijk aangekondigde Sun xVM Server-product", zegt Rich Green, executive vice president van Sun Software." "Daar waar Sun xVM Server is ontworpen dynamische IT te brengen in het hart van het datacenter, is VirtualBox ideaal voor elke laptop- of desktop-omgeving. Het sluit tevens perfect aan op Sun's andere producten gericht op de ontwikkelaars, zoals GlassFish, OpenSolaris, OpenJDK en binnenkort MySQL, evenals veel andere open source projecten. Hierdoor kunnen ontwikkelaars snel de nieuwste generatie applicaties ontwikkelen, testen en in gebruik nemen." De overname van innotek zal naar verwachting worden afgerond voor het eind van dit kwartaal.

## Apache Wicket 1.3

Apache Wicket 1.3 is vrijgegeven. Wicket is een Java open source component based web-framework, werkend met POJO's en HTML. Wicket maakt geen gebruik van XML tags. Het team richt zich op het produceren van geldige HTML en op een logische scheiding tussen ontwerp en code. Deze release is de eerste Apache-release. Hij is JDK-1.4 gebaseerd (de volgende zal Java5 zijn). Andere vernieuwingen zijn onder meer:

- Werkt nu zonder configuratie achter een proxy server gebruikmakende van relatieve URLs
- Google Guice support
- Wicket pages kunnen direct in een portal gebruikt worden (JSR-168/ JSR-286 ondersteuning)
- Logging API van commons-logging naar slf4j

Meer info http://wicketstuff.com

## Android-support voor Pulse / Eclipse

Genuitec (MyEclipse) heeft ondersteuning voor Google Adroid en Cold Fusion in het gratis product Pulse opgenomen. Naast ondersteuning voor Android en Cold Fusion is er ook ondersteuning voor veel andere open source en gratis tools toegevoegd. Zie ook ons eerdere bericht over Pulse.