Universele configuratie methode

Project 22

Door: ing. Jeffrey Barendse
Begeleider: Marten Hoekstra
Systeem en Netwerkbeheer
Universiteit van Amsterdam

5 februari 2005

Inleiding 2
Standaardisatie 3
Open standaarden 3
Commercie en open standaarden 3
Evolutie van standaarden 4
Conclusie 4
Systeem configuratie 5
Hosting provider 5
Koppeling met andere systemen 5
Configuratie methodes 6
Configuratie in de praktijk 7
Conclusie praktijkonderzoek 8
Configuratie interface 9
LDAP 9
SNMP 9
SOAP 10
Conclusie 10
Simpel Object Access Protocol 11
Begrippen 11
SOAP stack 12
Opbouw universeel configuratie systeem 13
Programmeer omgeving 14
Universeel beheer programma 14
Onderzoeksconclusie 15
Aanbevelingen 15
Bijlage A - Gentoo installatie op x86 Pentium4 16
Bijlage B - Gentoo e-mail installatie (spam, virus, webmail, mysql) 21
Bronvermeldingen 27

Inleiding

Systeembeheerders werken over het algemeen niet volgens een bepaalde methode. Zeker systeem beheerders die werken met OS3 systemen zijn geheel vrij in het installeren en configureren van de systemen naar hun eigen wens. Door veranderingen in versies of het aantrekken van extra systeembeheerders kan er een wirwar ontstaan van configuratie methodes. Door de colleges van Distributed Internet Applications ben ik geïnteresseerd geraakt in SOAP. Door gebruik te maken van SOAP verwacht ik dat het mogelijk is een methode te ontwikkelen die systeembeheer universeel kan maken. In dit onderzoek wil ik bekijken of het mogelijk is om een groot scala aan daemons en werkwijzen van systeembeheerders te koppelen tot één centrale infrastructuur die op een universele wijze te beheren is.

Standaardisatie

Open standaarden

Het gebruik van open standaarden heeft volgens velen alleen maar voordelen. Zo bevorderen open standaarden de toegankelijkheid van informatie. Door het gebruik van open standaarden kunnen verschillende softwaretoepassingen met elkaar communiceren en is het mogelijk om van softwareleverancier te wisselen zonder informatieverlies en zonder grote investeringen in conversie. Hiermee is de gebruiker verzekerd dat informatie in de toekomst toegankelijk blijft door de vrije toegankelijkheid van de specificaties.

Door gebruik van open standaarden zijn informatiestromen inzichtelijk voor de gebruiker. De gebruiker kan controleren welke informatie uit de toepassing komt en wie deze informatie ontvangt. Door de inzichtelijkheid van de informatiestromen kan de gebruiker de toepassing controleren op de beveiliging van de informatie en de betrouwbaarheid hiervan. De openheid van de specificaties maken het voor iedereen mogelijk om toekomstige verbeteringen voor te stellen.[I]

Commercie en open standaarden

Het gebruik van open standaarden heeft voor de wetenschap en eindgebruikers van software producten veel voordelen. Vanuit een commercieel oogpunt kunnen open standaarden ook nadelig zijn. Open standaarden maken het bijvoorbeeld mogelijk om eenvoudiger van softwareleverancier te wisselen. Iets wat de oorspronkelijke leverancier liever niet ziet gebeuren. Toch bieden veel commerciële softwareleveranciers hun diensten aan op basis van open standaarden omdat de afnemers open standaarden steeds meer als eis stellen.

Een mooi voorbeeld van een commerciële open standaard vind ik SQL. SQL99 is een gedefinieerde standaard die het mogelijk zou moeten maken om eenzelfde query uit te voeren op bijvoorbeeld een Oracle, MySQL of MSSQL database met hetzelfde resultaat. In de praktijk is dit helaas niet mogelijk. Uit commercieel oogpunt is het voor bedrijven niet wenselijk om geheel te voldoen aan SQL99 standaard. Er zou immers geen wezenlijk verschil meer zijn tussen de werking van producten waardoor een prijzenoorlog zou kunnen ontstaan. Zodra het commerciële product maar voldoende aan de standaard voldoet om dit op de verpakking te kunnen vermelen zal het bedrijf zich niet verder inspannen om geheel aan een standaard te voldoen. Aan de andere kant voldoen veel projecten als MySQL niet aan de gehele SQL99 standaard omdat er niet genoeg mankracht is om dit te verwezenlijken.[II]

Naast de commercie is ook “koppigheid” een gevaar voor standaarden. De Commercie komt zo nu en dan ook met oplossingen of ideeën waar nog geen gedefinieerde standaard voor is. Zo stelde Microsoft een routeringsstandaard voor SOAP/webservices berichten voor met de naam WS-Routing toen er nog geen andere concrete voorstellen waren. Omdat WS-Routing, om wat voor reden dan ook, niet geaccepteerd werd als standaard onderdeel van SOAP ontstond er meteen een keuze mogelijkheid voor ontwikkelaars waardoor het niet meer mogelijk was een standaard te bereiken die universeel routering mogelijk maakt.[III]

Evolutie van standaarden

Het ontwikkelen van standaarden neemt veel tijd in beslag. Een overeenkomst bereiken tussen verschillende partijen gaat nu eenmaal niet zonder slag of stoot. De Informatie Technologie is nog een vrij nieuw werkgebied. Nieuwe ideeën volgen elkaar snel op en daardoor ontstaat er ook een hogere tijdsdruk op standaarden om deze snel te ontwikkelen. Als een standaard pas voltooid is als de technologie eigenlijk al verouderd is zou al het werk voor niets geweest zijn. Omdat het niet goed mogelijk is om binnen korte tijd een standaard op te stellen die voor de meeste gevallen voldoet, ontstaan er vaak aftakkingen van een standaard. Zo maakt Oracle bijvoorbeeld gebruik van zijn eigen SQL taal als aanvulling op de standaard SQL taal. [IV]

Omdat IT standaarden eigenlijk niet voldoende of onduidelijk gespecificeerd zijn is het noodzakelijk dat bedrijven zich zo veel mogelijk aan een standaard houden. Een mooie lijdraad hierbij is het Internet Engineering Principle van Postel's. Volgens Postel is om IT standaarden goed te gebruiken het volgende noodzakelijk: "In general, an implementation must be conservative in its sending behavior, and liberal in its receiving behavior.". [V]

Conclusie

Standaarden zijn nog niet voldoende voor alle situaties of door commercie niet gewenst. Naar mijn mening is het ook onmogelijk om een standaard in één keer te realiseren. Het realiseren van een standaard is een tijd verslinden proces wat een evolutie doorgaat.
Met de tijd zullen standaarden evolueren naar standaarden die het mogelijk maken de gewenste doelen te bereiken. Denk hierbij aan één SQL query voor welke database dan ook.

Naar mijn mening is het geen goede zaak dat iedereen maar aftakkingen maakt op standaarden. Meer "halve" standaarden vergroot de complexiteit tot het bereiken van één definitieve standaard. In dit project zou ik bijvoorbeeld een aanpassing kunnen voorstellen voor LDAP of SNMP om het mogelijk te maken alle services op een computer via een universele manier te beheren. Hiermee bereik ik het tegenovergestelde van mijn doel, het eenvoudig koppelen van bestaande standaarden en eigen ideeën, omdat er veel gebruikers zullen zijn die hier de noodzaak niet van in zien en de standaard voor hun te complex wordt om nog eenvoudig te kunnen gebruiken/implementeren. Het probleem waar ik een oplossing voor wil vinden is dat standaarden vaak niet voldoen aan de wensen van een eindgebruiker maar het toch mogelijk moet zijn eenvoudige aanvullingen te maken op een standaard zonder de standaard te verbreken door het invoegen van eigen aanpassingen.

In dit onderzoek wil ik zoeken naar een mogelijkheid om op een standaard manier standaarden te kunnen koppelen/lijmen. Eigenlijk wil ik een methode ontwikkelen om standaarden te kunnen lijmen aan eigen ideeën. In dit onderzoek zal ik me hoofdzakelijk richten op het beheren van daemons die bedoeld zijn voor hosting providers.

Systeem configuratie

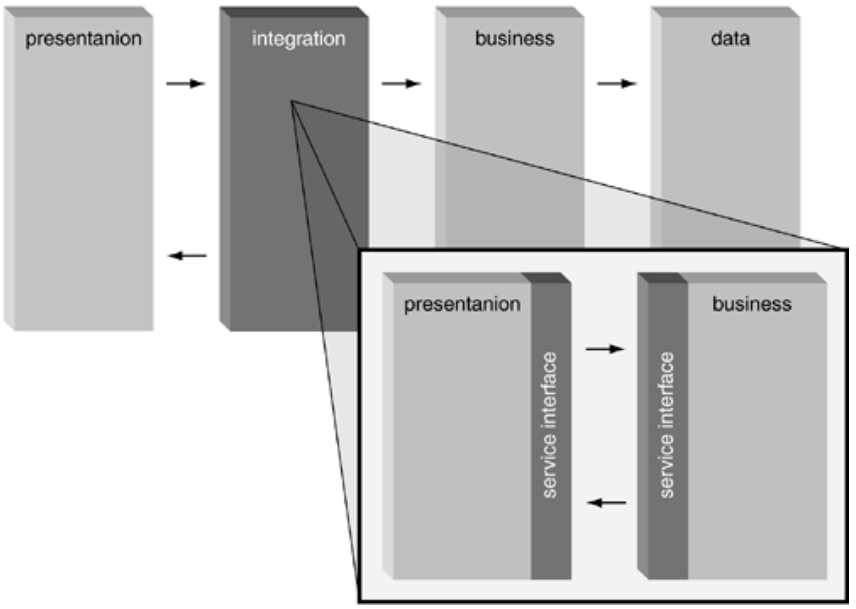
Hosting provider

Veel voorkomende diensten bij hosting providers zijn e-mail, website & database daemons. De hosting provider heeft keuze uit een scala aan leveranciers, merken en versies om deze diensten te kunnen realiseren. Naast de keuze die de hosting provider maakt kan elke daemon vaak geheel naar wens van de systeembeheerder geïnstalleerd worden. Denk hierbij aan installatiepaden en configuratiemethodes als een API of configuratie bestand. Ook is het niet ongebruikelijk dat een hosting provider diverse daemons aanbiedt. Zo kan de afnemer vaak zelf kiezen tussen een Linux of Windows webserver. Al deze keuzes maken het beheer van de hosting provider niet echt eenvoudiger. Ook het migreren naar andere daemons is vaak een groot probleem omdat ze totaal niet compatible met elkaar zijn.

Koppeling met andere systemen

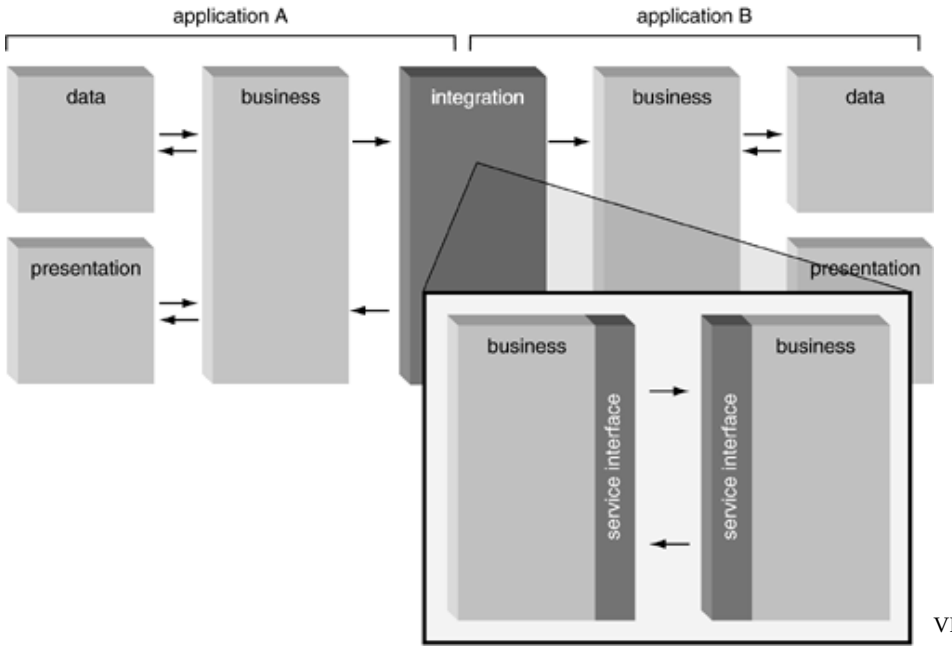
Naast het configureren van daemons kan het ook wenselijk zijn de diensten te koppelen aan een facturatie systeem. Aangezien er zeer veel configuratie methodes en mogelijkheden zijn is dit niet eenvoudig te realiseren. Ook een verandering aan één van de onderdelen van het systeem zou beteken dat er op meerdere plaatsen aanpassingen gemaakt moeten worden om alles weer te laten functioneren. Ik wil onderzoeken wat de mogelijkheden zijn om een dergelijk probleem gestructureerd en gestandaardiseerd aan te pakken.

Om een dergelijk systeem muli-tier te realiseren moet er integratie laag gebouwd worden tussen de presentatie (factuur) en de business (daemon) laag. Deze integratie laag moet beide talen (presentatie en business laag) spreken. Een aanpassing kan gevolgen hebben voor meerdere lagen. Iets wat het systeem een stuk complexer te beheren maakt. Een oplossing hiervoor is een service-oriented[VI] architecture (SOA). Hierbij wordt de integratie laag vervangen door een gestandaardiseerd protocol in zowel de presentatie als business laag (service interface). Een voorbeeld van een dergelijk gestandaardiseerd protocol is een webservice. Dit maakt de applicatie minder gevoelig voor aanpassing, maar de afhankelijkheid tussen de presentatie en business laag blijft bestaan (zie afbeelding).



[VII]

Om de afhankelijkheid tussen de applicaties te verhelpen is het noodzakelijk om een service-oriented integration architecture te gebruiken. Hiermee is de applicatie een zelfstandige entiteit die kan samenwerken met andere entiteiten via een gestandaardiseerd protocol. In het geval van een hosting provider betekend dit dat het mogelijk is een infrastructuur op te bouwen uit logische bouwstenen. Elke server is een eigen entiteit/bouwsteen die met andere entiteiten kan communiceren (zie afbeelding).



VIII

Configuratie methodes

Om tot een methode te komen om het beheren van diverse daemons en het koppelen van eigen/andere systemen/standaarden eenvoudiger te maken heb ik eerst in kaart gebracht wat de methodes zijn om daemons te configureren/beheren. Om ervaring op te doen met diverse daemons en hun bijbehorende configuratie methodes heb ik er voor gekozen een test opstelling te creëren met de volgende software:

- **Gentoo Linux**
  Een Linux variant waarbij de gebruiker zelf het OS samenstelt en dus veel vrijheid heeft in de keuzes die hij maakt. Mijn keuze is op Gentoo gevallen omdat dit in mijn ogen een goede eigenschap is voor mijn test opstelling.[IX]

- **Qmail**
  Een snelle, betrouwbare, veilige, eenvoudige en de snelst groeiende e-mail deamon.[X]

- **Vpopmail**
  Een configuratie programma die het beheer van virtuele e-mail domeinen eenvoudiger maakt. Deze daemon zal gekoppeld worden met Qmail.[XI]

- **MySQL**
  De meest gebruikte opensource database ter wereld. Een database is noodzakelijk om vpopmail te laten functioneren.[XII]

- **Courier IMAP**
  Een daemon voor het ophalen van e-mail via IMAP en POP. Deze daemon zal gekoppeld worden met Qmail voor de authenticatie.[XIII]

- **SpamAssassin**
  Een mailfilter die spam identificeert op basis van regels en een punten systeem. Deze gaat gekoppeld worden aan Qmail om automatische alle inkomende e-mail te scannen.[XIV]

- **Clam Antivirus**
  Een virusscanner die gebruikt kan worden om e-mail te scannen. Deze gaat gekoppeld worden aan Qmail om automatisch alle inkomende e-mail te scannen.[XV]

- **Horde IMP**
  Het applicatie framework van Horde biedt diverse applicaties waaronder een webmail client genaamd IMP. Deze zal gekoppeld worden met Qmail en vpopmail[XVI]

- **Apache**
  De meeste gebruikte webserver ter wereld. Hierop zal het Horde framework draaien[XVII]

Door de keuze voor deze combinatie van software en daemons die met elkaar geïntegreerd moeten worden verwacht ik veel informatie uit de test te kunnen krijgen met betrekking tot de configuratie mogelijkheden van daemons.

Configuratie in de praktijk

Tijdens mijn praktijktest heb ik een werkende testopstellingen kunnen creëren met een koppeling tussen de daemons. In bijlage A en B heb ik de mijn handleidingen opgenomen voor het installeren en configureren van deze daemons voor de test opstelling. Het was erg opvallend dat ik werkelijk geen enkele handleiding voor één van de onderdelen van deze praktijktest heb kunnen vinden zonder fouten. De handleidingen kloppen vaak niet omdat er andere versies gekoppeld zijn, een andere werkwijze gebruikt wordt of simpelweg zaken vergeten zijn. Ik heb dit probleem opgelost door veel informatie over bepaalde daemons op te zoeken zodat ik een goed beeld had van de werking. Hierdoor was ik zelf in staat fouten in handleidingen op te sporen en te verbeteren in mijn eigen handleiding. Toch is er veel tijd gaan zitten in het creëren van een werkende test opstelling door deze fouten in handleidingen. Ik ben van mening dat standaarden en handleidingen dit probleem delen. Er zijn zoveel mogelijkheden, qua configuratie in dit geval, dat het niet mogelijk is overal een handleiding/standaard voor te creëren.

De daemons die ik onderzocht heb gaven mij een helder beeld van de configuratie methodes. Ik heb vier methodes kunnen vinden voor het beheren van een daemon. Uit mijn onderzoek/testopstelling maak ik op dat alle daemons met één van de volgende methodes te configureren zijn:

- **Configuratie bestand(en)**
- **Command's vanaf de commandline**
- **Database query's**
- **API's**

Ik heb geen andere daemons kunnen bedenken die niet op deze manier te configureren zijn. Opvallend is dat diverse daemons op meerdere manieren te configureren zijn. Zo is vpopmail zowel te beheren via de commandline als een API.

Door de wijze waarop ik alle software geïnstalleerd en geconfigureerd heb is het nu mogelijk om met enkele commandline command's een e-mail account (of via de API) aan te maken. Het resultaat is een werkende e-mail box incl. aliassen met zowel POP als IMAP en een webmail cliënt. Al deze diensten kunnen zowel met als zonder SSL gebruikt worden. Ik heb gekozen voor deze configuratie omdat dit het ontwikkelen van een voorbeeld applicatie voor mijn methode eenvoudiger maakt. Ik kan me dan richten op de werkwijzen van de applicatie in plaats van een uitgebreid programma schrijven wat alle configuratie bestanden kan aanpassen en verifiëren. In verband met de korte duur van dit project is dit noodzakelijk om voortgang te boeken binnen de gestelde termijn van 1 maand. Ook is het qua functie omschrijving van een systeembeheerder noodzakelijk een oplossing te vinden die zo min mogelijk programmeerwerk vereist. Deze werkwijze maakt de broncode van het programma een stuk eenvoudiger, er hoeft alleen maar een commando aangeroepen te worden, waardoor de kans op fouten afneemt.

Conclusie praktijkonderzoek

Op basis van mijn praktijkonderzoek ben ik tot de volgende conclusie gekomen. Om alle daemons op een gestandaardiseerde wijze te kunnen beheren is een interface (business laag) noodzakelijk die de daemons kan aanspreken op één of meerdere van de onderstaande methode. Deze interface moet via een service interface met andere entiteiten kunnen communiceren om onafhankelijkheid van andere systemen te creëren. De interface is eigenlijk een vertaler tussen de configuratie methode en de service interface (zie afbeelding).

- **Bestanden**
  - aanmaken (met bepaalde rechten)
  - bewerken (variabelen opzoeken/aanpassen/etc)
  - verwijderen

- **Commando**
  - commando uitvoeren (runnen als bepaalde gebruiker)

- **Database query**
  - SQL query uitvoeren (als bepaalde gebruiker)

- **API**
  - programma interface (directe koppeling)

Een bijkomend resultaat wat niet tot dit onderzoek behoord is de lage kwaliteit van de handleidingen van het gros van de opensource projecten. Dit is mogelijk een interessante onderzoeksvraag voor een ander project.

Configuratie interface

De interface tussen de daemon configuratie en de andere applicaties kan op meerdere manieren gerealiseerd worden. Op basis van mijn vooronderzoek is mijn keuze gevallen op drie mogelijkheden.

LDAP

In eerste instantie viel mijn keuze op LDAP (Lightweight Directory Access Protocol). LDAP is een client/server omgeving waardoor het niet mogelijk is om elk systeem te laten functioneren als een aparte entiteit. [XVIII] Ik heb geen specifieke eigenschappen van LDAP gevonden die een voordeel kunnen zijn voor dit project. Enkele nadelen om LDAP te gebruiken als een configuratie interface vind ik:

- **Complexiteit van de LDAP database**
  Niet geschikt om snel een aanpassing te kunnen realiseren.

- **LDAP geoptimaliseerd voor leesacties**
  Iets wat ik ongewenst vind aangezien de gebruiker van de interface zelf moet kunnen bepalen wat hij wil lezen of schrijven in welke richting.

- **Afhankelijkheid van de LDAP server**
  Ik wil geen onnodige afhankelijkheid creëren om het systeem zo modulair en eenvoudig mogelijk te houden.

SNMP

Een andere methode zou SNMP (Simple Network Management Protocol) kunnen zijn. Dit protocol is gemaakt voor zowel lees als schrijfbewerkingen. De opbouw is, net als LDAP, een client/server structuur. [XIX] SNMP is echter niet afhankelijk van een server. Ik heb geen specifieke eigenschappen van SNMP gevonden die een voordeel kunnen zijn voor dit project. Enkele nadelen om SNMP te gebruiken als een configuratie interface vind ik:

- **Versie verschillen**

SNMP zit vaak “ingebakken” in hardware waardoor er versieverschillen ontstaan. Bijvoorbeeld geen ondersteuning voor SSL wat wel wenselijk is.

- **SNMP niet over server aanwezig**

De SNMP software is vaak niet op servers aanwezig. Dit vraag de nodige werkzaamheden om SNMP werkend te krijgen.

- **MIB (Management Information Base) niet altijd beschikbaar**

Voor lang niet alle hardware/software is een MIB beschikbaar. Deze moet dan gerealiseerd worden volgens de standaard. Dit is een secuur en complexe aangelegenheid die aanpassingen achteraf niet goed mogelijk maakt. Een aanpassing in één van de MIB’s zou als gevolg hebben dat alle applicaties aangepast moeten worden om met de aangepaste MIB te kunnen werken. Zelfs als een MIB wel beschikbaar is zijn deze over het algemeen fabrikant afhankelijk. Het is dus niet mogelijk om volgens een universele methode de hardware/software te beheren. [XX]

- **Geen RPC (Remote Procedure Call)**

Het protocol is bedoeld voor lees/schrijf bewerkingen. De interface zou dus moeten "pollen" of er aanpassingen zijn in de MIB om tot daadwerkelijke actief over te gaan. Dit maakt interactie met het systeem een stuk complexer.

- **Geen geschikte foutafhandeling**

SNMP maakt gebruik van "fire and forget". Het is dus onduidelijk wat er met opdrachten precies gebeurd. Elke opdracht zou dus achteraf geverifieerd moeten worden ter controle.

SOAP

De methode waar ik voor gekozen heb is SOAP (Simpel Object Access Protocol). Deze methode is geschikt voor het creëren van applicaties volgens de SOA. Net als bijvoorbeeld CORBA (Common Object Request Broker Architecture) is SOAP bedoelt voor het creëren van gedistribueerde applicaties. In tegenstelling tot methodes als CORBA is SOAP platform/programmeertaal onafhankelijk. De belangrijkste voordelen van SOAP vind ik:

- **Communicatie via HTTP (Hyper Text Transfer Protocol)**

SOAP biedt de mogelijkheid om via HTTP te communiceren met andere service interfaces. Aangezien eigenlijk alle servers wel over TCP beschikken is de implementatie van HTTP eenvoudig. Een groot voordeel van HTTP is dat firewalls dit meestal niet blokkeren. Ook is het mogelijk om SSL (Secure Socket Layer) toe te voegen aan HTTP. Hierdoor is de beveiliging op een transparante manier te regelen voor het systeem.

- **Berichten uitwisseling in XML**

Tussen de service interfaces gaat de communicatie over HTTP. De berichten die uitgewisseld worden zijn in het XML formaat. In tegenstelling tot bijvoorbeeld CORBA is dit een formaat wat voor een mens begrijpelijk is. Dit maak het opsporen van fouten een stuk eenvoudiger.

- **Automatische herkenning van services**

Webservices op basis van SOAP bieden de mogelijkheid diensten automatisch te herkennen en te kunnen gebruiken.

Conclusie

Om een universeel systeem te realiseren voor het configureren van daemons is het noodzakelijk een service-oriented integration architecture te creëren waarbij de applicaties gekoppeld worden door middel van een service interface. Door te kiezen voor SOAP voor de service interface is het mogelijk een platform/programmeertaal onafhankelijke entiteit te creëren. Elk systeem kan onafhankelijk van elkaar werken maar desgewenst ook communiceren of samenwerken met andere systemen.

De service interface zal communiceren met de business laag op de entiteit. De business laag zal zorg dragen voor het uitvoeren van de vier configuratie methodes: bestanden, commando, database en API. Deze methodes zijn platform, programmeertaal en daemon afhankelijk. De service interface dient als vertaler tussen de business laag en de andere entiteiten. Hierdoor is het mogelijk om een universele wijze alle daemons te kunnen configureren.

Simpel Object Access Protocol

Om een werkend test opstelling te creëren is het noodzakelijk dat ik een applicatie ontwikkel die op basis van SOAP onderling kan communiceren met de andere entiteiten en daarnaast kan communiceren met de daemons middels een van de vier configuratie methodes.

Om een goed beeld te krijgen van de mogelijkheden en eigenschappen van SOAP heb ik me verdiept in het programmeren van webservices met SOAP.[XXI] De belangrijkste mogelijkheden en eigenschappen zal ik in dit hoofdstuk kort samenvatten.

Begrippen

- **WSDL (Web Services Description Language)**

Een protocol wat de diensten op een bepaalde entiteit kan beschrijven. WSDL is te vergelijken met een functie declaratie in een programmeertaal. Door het opvragen van de WSDL van een entiteit kan een andere entiteit bepalen wat de dienst inhoudt en hoe deze werkt.

- **UDDI (Universal, Description, Discovery and Integration)**

Een protocol wat automatische herkenning van entiteiten en bijbehorende diensten mogelijk maakt. UDDI is te vergelijken met een soort "Goudengids" voor diensten. Dit kan gebruikt worden om automatisch alle diensten in een netwerk te herkennen. Een WSDL entiteit meldt zich aan bij de UDDI om opgenomen te worden in de "Goudengids".

- **Service Listener**

SOAP is eigenlijk een XML bericht met een vastgestelde opmaak. Het XML bericht wordt verzonden over TCP/IP via HTTP. Op beide entiteiten is een Service Listener nodig die de berichten kan ontvangen en verzenden. Dit zou bijvoorbeeld een webserver kunnen zijn als Apache. SOAP is dus geen op zich zelf staand protocol maar afhankelijk van anderen.

- **Service Proxy**

De berichten tussen de programmeertaal en de Service Listener moeten omgezet worden van en naar SOAP. Om de programmeur hier niet mee te belasten is er de Service Proxy. Dit is over het algemeen een bibliotheek die de omzetting regelt. Het is ook mogelijk op basis van een WSDL een Service Proxy te generen. Stel dat er een entiteit is die wiskundige berekeningen kan maken. Als er op een andere entiteit een Service Proxy gegenereerd wordt op basis van de WSDL van de wiskundige entiteit, wordt automatisch de Service Proxy ingesteld voor de juiste functies/variabelen en de bijbehorende broncode gegenereerd.

SOAP stack

De SOAP stack is opgebouwd uit vijf lagen. Het is mogelijk verschillende methodes te gebruiken per laag. Ik heb gekozen voor de volgende methode:

- **Discovery - UDDI**

Deze laag maakt het mogelijk om de diensten te adverteren. In mijn onderzoek zal ik verder geen aandacht besteden aan UDDI omdat dit in verband met de tijdsdruk niet mogelijk is. Aangezien UDDI een laag is die boven op WSDL komt is het later toevoegen van UDDI mogelijk. Ook is UDDI naar mijn mening niet in alle gevallen noodzakelijk of gewenst.

- **Description – WSDL**

Om de entiteiten volgens een universele methode met elkaar te laten communiceren is het noodzakelijk dat de entiteiten van elkaar te weten kunnen komen hoe ze werken. WSDL is als het ware een gebruikshandleiding voor de andere entiteiten.

- **Packaging - XML**

De communicatie tussen de entiteiten zal gebeuren in het XML formaat.

- **Transport - TCP/HTTPS**

De connectie tussen de entiteiten zal opgebouwd worden op basis van TCP. Een HTTP daemon zal zorg dragen voor de afhandeling van de in en uitgaande XML pakketen. De HTTP daemon is in dit geval de service listener.

- **Network - Basis TCP/IP layer**

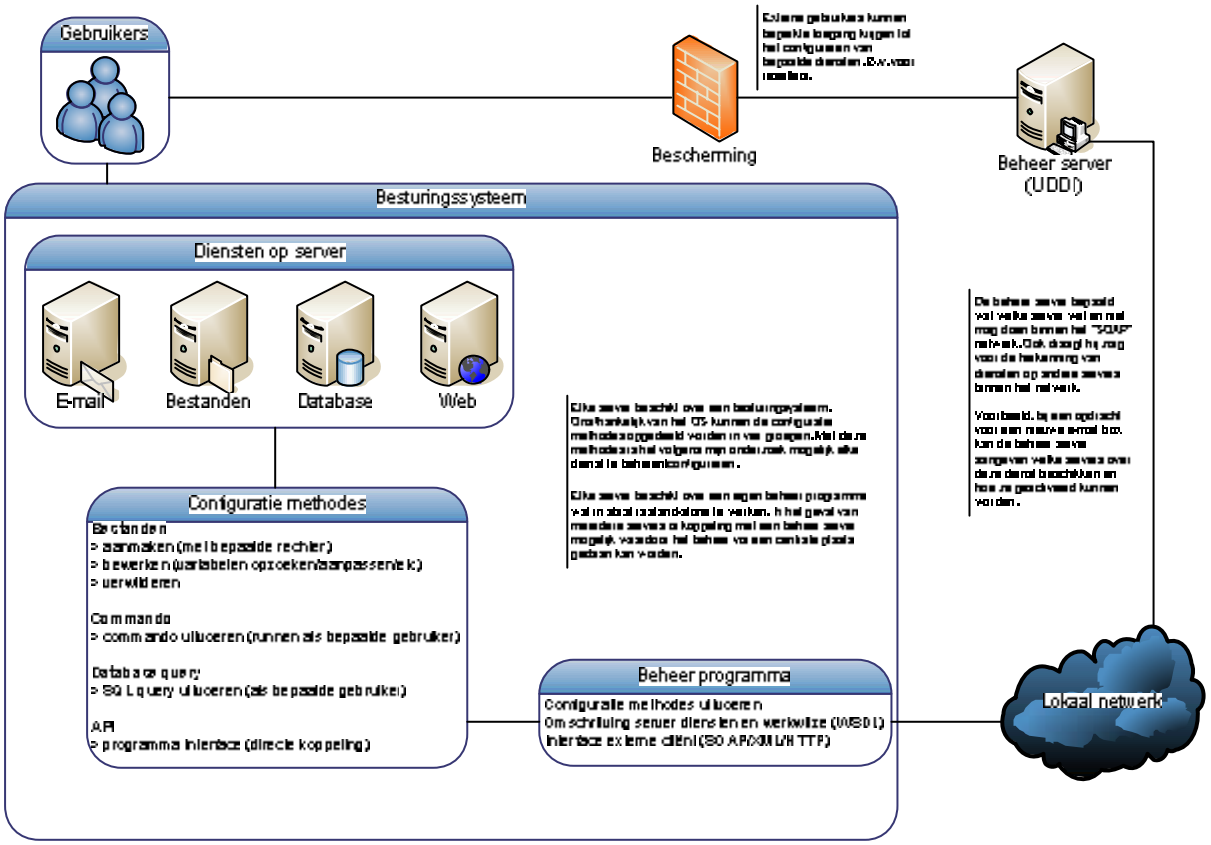
Deze laag is gelijk aan de TCP/IP laag omdat we gebruik maken van TCP/IP als protocol.

Opbouw universeel configuratie systeem

Op basis van mijn onderzoek kom ik tot de volgende opzet voor het realiseren van een infrastructuur die op een universele wijzen te beheren is:

- Daemons moeten geïnstalleerd worden op de server
- Configuratie methodes moeten gebouwd worden voor de specifieke daemon
- Een beheer programma moet de koppeling worden tussen de configuratie methodes en SOAP. Deze beheer applicatie zal zijn diensten aangeven in het WSDL formaat.
- Een centrale server zal dienen als manager. Op basis van UDDI adverteert deze de diensten aan de beheer programma's op verschillende entiteiten.

Om een duidelijker beeld te krijgen van de opbouw van het configuratie systeem heb ik het onderstaande schema gemaakt.



Programmeer omgeving

De beheer applicatie kan in diverse programmeer omgevingen ontwikkeld worden. Ik heb diverse omgevingen onderzocht.

- **PHP & NuSOAP**
  PHP in combinatie met NuSOAP maakt het mogelijk om op een eenvoudige wijze SOAP en WSDL te implementeren. Om hiervan gebruik te maken is het wel noodzakelijk een webserver als Apache op de entiteit te installeren.[XXII]

- **Perl & SOAP::Lite**
  Deze combinatie maakt het mogelijk om op bijna elk Unix platform meteen aan de slag te kunnen. Bijna alle Unix varianten beschikken standaard over Perl. Perl is een geïnterpreteerde taal die niet stand-alone kan draaien. Net als bij PHP is er voor Perl een CGI server nodig als Apache om SOAP te laten functioneren.[XXIII]

- **Java**
  Ook Java biedt diverse bibliotheken aan voor SOAP. Java is een geïnterpreteerde taal die niet altijd standaard aanwezig is op een server en handmatig geïnstalleerd dient te worden. Java kan zowel stand-alone als via een webserver gebruikt worden.

- **C++**
  In eerste instantie lijkt C++ niet de perfecte taal voor een dergelijk systeem omdat dit toch een complexere programmeertaal is dan de andere opties. Voor C++ ben ik een framework tegen gekomen genaamd Gsoap. Dit is een complete SOAP omgeving voor C++ die beschikt over een optionele ingebouwd HTTP server met SSL. Hierdoor is het mogelijk een compleet zelfstandige applicatie te creëren die geen afhankelijkheid heeft. De Gsoap bibliotheek is portable naar bijna alle platformen wat gebruik van diverse besturingssystemen mogelijk maakt.[XXIV]

De test applicatie zal ik ontwikkelen in Gsoap. De eigenschappen van Gsoap passen het beste bij het gewenste eindresultaat: een universele configuratie methode met zo min mogelijk afhankelijkheid.

Universeel beheer programma

De structuur en opbouw van de universele beheer methode zijn inmiddels gevormd. Als voorbeeld heb ik een eenvoudige SOAP applicatie in Gsoap ontwikkeld. De applicatie die ik gemaakt heb is bedoel als een eerste test voor mijn universele beheer methode. Deze applicatie beschikt over:

- Ingebouwde HTTP daemon (geen andere daemons nodig)
- SSL met encryptie en authenticatie door middel van cliënt & server certificaten

Onderzoeksconclusie

Op basis van mijn bevinden en aanbevolen opbouw moet het mogelijk zijn een universele beheer methode te kunnen realiseren. In verband met de korte duur van dit onderzoekproject van minder dan een maand is het helaas niet mogelijk een compleet werkend prototype op te leveren voor alle daemons. Ik ben gekomen tot een werkende applicatie die op basis van SOAP RPC berichten kan uitwissel en waarbij authenticatie en encryptie geregeld is om basis van SSL certificaten. Hiermee heb ik de echte problemen "getackeld" en kan ik me in een mogelijk vervolgonderzoek bezig houden met het realiseren van de daadwerkelijke code voor het uitvoeren van de vier configuratie methodes. Ondanks dat het niet mogelijk is een complete applicatie te realiseren ben ik erg tevreden met het resultaat en wil ik deze methode verder ontwikkelen en uitwerken.

Aanbevelingen

Voor een vervolg onderzoek zou ik de volgende zaken verder willen uitwerken:

- **Uitgebreidere authenticatie methode**

  De huidige SSL certificaten in combinatie met HTTP autenticatie bieden een eenvoudige rechtenstructuur. Om de ontwikkelaar niet te veel te belasten met beveiligingsregels tijdens het programmeren is het wenselijk gebruik te maken van een standaard framework voor dergelijke zaken. De mogelijkheid tot het implementeren van GSI (Grid Security Infrastructure)

- **Realisatie prototype applicatie**

  Om het systeem daadwerkelijk te testen is het noodzakelijk dat er een werkende applicatie gerealiseerd wordt. Hiermee kan het theoretische idee van een universele beheer methode praktisch getoetst worden.

Bijlage A - Gentoo installatie op x86 Pentium4

Download de CD-rom

- Download ISO met minimum livecd
- Brand ISO
- Boot van CD

Configureren netwerk

- Het system detecteert automatisch een DHCP server.

Partities aanmaken

- Fdisk starten voor de juiste harddisk
  > fdisk /dev/hda
- Bekijk partitie tabel:
  > p
- Verwijder bestaande partities nummer voor nummer:
  > d 4
- Maak nieuwe partities aan (n, p, 1, 1, +50MB, etc.)
  - /boot (/dev/hda1, 50M)
  - Swap (/dev/hda2, 1G) zet ID op 82 voor SWAP
  - / (/dev/hda3, rest van HDD)
- Bewaar partitie tabel
  > w
- Sluit Fdisk en herstart systeem
  > q
  > reboot

Initialize

- /boot to ext3:
  > mke2fs –j /dev/hda1 –m0 (geen space voor root reserveren)
- / to ext3:
  > mke2fs –j /dev/hda3
- Initialize and activate swap:
  > mkswap /dev/hda2
  > swapon /dev/hda72

Mount

- /dev/hda3
  > mount /dev/hda3 /mnt/gentoo
- /dev/hda1
  > mkdir /mnt/gentoo/boot
  > mount /dev/hda1 /mnt/gentoo/boot
- Bind mounten /dev
  > mkdir /mnt/gentoo/dev
  > mount –o bind /dev /mnt/gentoo/dev

Gentoo installatie Stage 3 bestanden

- Contole datum en tijd i.v.m. synchronisatie Portage
  > date
  > date MMDDhhmmYYYY
  > cd /mnt/gentoo
- Download stage3 & Portage snapshot Portage op: (b.v. Mirror > X86 > 2004.3 > Stages > P4 > Stage3)
  > links2 http://www.gentoo.org/main/en/mirors.xml
  > tar –xvjpf stage3*.tar.bz2
  > tar –xvjf portage*.tar.bz2 –c /mnt/gentoo/usr
- Pas make.conf aan voor juiste processor
  > vi /mnt/gentoo/etc/make.conf

  # These settings were set by the catalyst build script that automatically built this stage
  # Snelste files O3, Allen P4 march, pipe ipv tmp file, 2 cpu -j2
  CFLAGS="-O3 -march=pentium4 -pipe -fomit-frame-pointer"
  CHOST="i686-pc-linux-gnu"
  CXXFLAGS="${CFLAGS}"
  MAKEOPTS="-j2"
  SYNC="rsync://rsync.europe.gentoo.org/gentoo-portage"
  GENTOO_MIRRORS="http://ftp.snt.utwente.nl/pub/os/linux/gentoo"

- Kopieer DNS info:
  > cp –L /etc/resolv.conf /mnt/gentoo/etc/resolv.conf
- Mount Proc:
  > mount –t proc none /mnt/gentoo/proc
- Start nieuwe opgeving:
  > chroot /mnt/gentoo /bin/bash
  > env-update
  > source /etc/profile
- Portage tree downloaden/updaten:
  > emerge --sync

Configureren van de kernel

- Stel timezone in:
  > ln –sf /usr/share/zoneinfo/Europe/Amsterdam /etc/localtime
- Installeer kernel source:
  > emerge gentoo-dev-sources
- Menuconfig:
  > cd /usr/src/linux
  > make menuconfig
    - Code maturity level options:
      - [*] Prompt for development and/or incomplete code/drivers
        - [*] Select only drivers expected to compile cleanly
    - Filesystems:
      - <*> Ext3 journalling file system support
      - [*] Ext3 extended attributes
      - <*> Kernel automounter version 4
      - [ ] DOS/FAT/NT Filesystem
      - Pseudo filesystems:
        - [*] /dev file system support
        - [*] Automatically mount at boot
    - Bus options
      - [ ] ISA support
    - Processor type and features
      - [*] Symmetric multi-processor support
      - (4) SMT (Hyper threading) scheduler support
    - Device driver:
      - [ ] SCSI device support:
      - [ ] IEEE 1394 (FireWire support)
      - Block devices
        - [ ] Normal floppy disk support
- Compileren kernel:
  > make && make modules_install
- Installeren Kernel:
  > cp arch/i386/boot/bzImage/bzImage /boot/kernel-2.6.10
  > cp System.map /boot/System.map-2.6.10
  > cp .config /boot/config-2.6.10

FSTAB

- Edit fstab:
  > nano –w /etc/fstab

/dev/hda1 /boot	ext2	defaults,noatime		1 1
/dev/hda3 /	ext3	noatime		1 1
/dev/hda2 none	swap	sw		0 0
/dev/cdroms/cdrom0	/mnt/cdrom	auto	noauto,ro	0 0
None	/dev/shm	tmpfs	nodev,nosuid,noexec	0 0

Network

- Hostname:
  > echo servernaam > /etc/hostname

- Domainnaam:
  > echo domein.nl > /etc/dnsdomainname
- Voeg domeinname toe aan runlevel
  > rc-update add domainname default
- Voeg hostname toe aan hosts file op problemen DHCP naam te voorkomen
  > vi /etc/hosts

  127.0.0.1 localhost hostnaam

- Netwerk configuratie:
  > nano –w /etc/conf.d/net
  iface_eth0=”dhcp”
- Voeg network toe aan runlevel:
  > rc-update add net.eth0 default

System

- Root passwd instellen:
  > passwd

Bootloader

- Installeren en configureren bootloader

> emerge grub
> nano /boot/grub/grub.conf

```
default 0
timeout 5
splashimage=(hd0,0)/grub/splash.xpm.gz

title=Gentoo Linux 2.6.10-rc5
root (hd0,0)
kernel /boot/kernel-2.6.10 root=/dev/hda3
```

> cp /proc/mounts /etc/mtab
> grub-install --root-directory=/boot /dev/hda

Extra pakketen installeren:

Emerge commando opties
-U = newest versie (somss beta)
-u = best stable version
-d = dependency check
-k = download binary if possible
--pretend = doen alsof en geeft lijst met te maken aanpassing

emerge metalog	(geavanceerde syslog)
rc-update add metalog default	
emerge vixie-cron	(cron demon)
rc-update add vixie-cron default	
emerge slocate	(locate commando)
emerge dhcpcd	(dhcp client)
Emerge vim	(vi editor)
Emerge screen	(remote ssh screen die actief blijft)

rc-update add sshd default

screen (remote shell die actief blijft)
screen –r (voor reopening activate shell)

Bijlage B - Gentoo e-mail installatie (spam, virus, webmail, mysql)

- Voeg modules toe aan /etc/make.conf voor automatische compilatie met pakketen

  USE="maildir ssl imap mysql nls"

- Verwijder huidge mail software (geïnstalleerd tijdens metalog) en installeer qmail

> emerge –unmerge ssmtp
> emerge qmail

- Instellen certificaat en basis config qmail laden:

> vi /var/qmail/control/servercert.cnf

  C=NL
  ST=Noord-Holland
  L=Wormerveer
  O=WireITup
  CN=hostnaam.domeinnaam.nl
  emailAddress=j.barendse@wireitup.nl

> ebuild /var/db/pkg/mail-mta/qmail-1.03-r*/qmail-1.03-r*.ebuild config

- Instellen standard aflever adressen voor standaard Qmail adressen.

> cd /var/qmail/alias
> echo mailmaster@wireitup.nl > .qmail-mailer-daemon
> echo mailmaster@wireitup.nl > .qmail-postmaster
> echo mailmaster@wireitup.nl > .qmail-root

- Koppelen qmail serverice en toevoegen aan runlevel

> rc-update add svscan default
> /etc/init.d/svscan start
> cd /service
> ln -s /var/qmail/supervise/qmail-send qmail-send

- Test of huidige host juiste domeinnaam config heeft

> hostname –fqdn

- Controleer op bovenstaande domeinnaam in volgende files staat (zo niet toevoegen)

> cd /var/qmail/control
> cat me
> cat defaultdomain
> cat plusdomain
> cat locals
> cat rcpthosts

- MySQL installer & configureren

> emerge mysql
> ebuild /var/db/pkg/dev-db/mysql-4.0.22/mysql-4.0.22.ebuild config
> /usr/bin/mysqladmin –u root –h 127.0.0.1 password ‘wachtwoord’
> rc-update add mysql default

- Vpopmail database aanmaken en config instellen

> vi /etc/vpopmail.conf

localhost|0|vpopmailr|1234|vpopmail
localhost|0|vpopmailw|12345|vpopmail

> mysql –u root -p
> create database vpopmail;
> use mysql;
> grant select on vpopmail.* to
vpopmailr@localhost identified by '1234';
> grant select, insert, update, delete, create, drop on vpopmail.* to
vpopmailw@localhost identified by '12345';
> flush privileges;
> quit;

- Rechten vpopmail configuratie bestanden instellen

> chown root:vpopmail /etc/vpopmail.conf
> chmod 640 /etc/vpopmail.conf
> chown root:vpopmail /var/vpopmail/bin/vchkpw
> chmod 4711 /var/vpopmail/bin/vchkpw

- Domein en gebruiker aanmaken in vpopmail

> source /etc/profile
> vadddomain wireitup.nl 1234
> vadduser j.barendse@wireitup.nl 12345

- Courier pop/imap installeren

> emerge net-mail/courier-imap
> cd /etc/courier-imap
> vi pop3d.cnf & imapd.cnf

C=NL
ST=Noord-Holland
L=Wormerveer
O=WireITup
CN=hostnaam.domeinnaam.nl
emailAddress=j.barendse@wireitup.nl

> mkpop3dcert
> mkimapdcert

- Courier demons opstarten toevoegen aan runlevel

> rc-update add courier-pop3d-ssl default
> rc-update add courier-pop3d default
> rc-update add courier-imapd-ssl default
> rc-update add courier-imapd default
> /etc/init.d/courier-pop3d-ssl start
> /etc/init.d/courier-pop3d start
> /etc/init.d/courier-imapd-ssl start
> /etc/init.d/courier-imapd start

- SMTP instellen en activeren

> cd /var/qmail/control/
> vi conf-smtpd (uncommend de volgende SMTP AUTH regels en pas aan)

```
QMAIL_SMTP_AUTHHOST=$(<${QMAIL_CONTROLDIR}/me)
[ -z "${QMAIL_SMTP_POST}" ] && QMAIL_SMTP_POST=/bin/true
QMAIL_SMTP_CHECKPASSWORD="/var/vpopmail/bin/vchkpw"
QMAIL_SMTP_POST="${QMAIL_SMTP_AUTHHOST}
${QMAIL_SMTP_CHECKPASSWORD} ${QMAIL_SMTP_POST}"
```

> cd /service
> ln -s /var/qmail/supervise/qmail-smtpd qmail-smtpd
> /etc/init.d/svscan restart

- Horde installeren en instellen

> emerge horde-imp
> cd /var/www/localhost/htdocs/horde/config
> for f in *.dist ; do mv ${f} ${f/.dist} ; done
> vi horde.php

```
$conf['auth']['driver'] = 'imap';
$conf['auth']['params']['dsn'] = '{localhost:993/imap/ssl/novalidate-cert}';

$conf['log']['name'] = '/var/log/apache2/horde.log';
```

> vi registry.php

```
$this->registry['auth']['login'] = 'imp';
$this->registry['auth']['logout'] = 'imp';

Set the 'status' element of applications['imp'] from 'inactive' to 'active'
```

> mkdir /var/cache/apache2
> mkdir /usr/lib/apache2/conf/ssl
> touch /var/log/apache2/horde.log
> chown –R apache:apache /var/log/apache2

> cd /var/www/localhost/htdocs/horde/imp/config/
> for f in *.dist ; do mv ${f} ${f/.dist} ; done
> vi servers.php

```
$servers['imap'] = array(
   'name' => 'wireitup.nl',
   'server' => 'localhost',
   'protocol' => 'imap/ssl/novalidate-cert',
   'port' => 993,
   'folders' => '',
   'namespace' => 'INBOX',
   'maildomain' => 'example.com',
   'smtphost' => 'localhost',
   'realm' => '',
   'preferred' => ''
);
```

Let op de hostname die Apache krijgt moet FQDN zijn anders werkt de client login niet (404). Normaal krijgt Apache deze naam vanzelf. Als FQDN op Gentoo niet goed staat kun je het verhelpen met de ServerName variable in /etc/apache2/conf/apache2.conf file.

- Spam en virus filtering

```
> emerge mail-filter/spamassassin
> vi /etc/spamassassin/local.cf

    rewrite_header Subject *****SPAM*****
    required_hits 2
    report_safe 0

> emerge app-antivirus/clamav
> vi /etc/conf.d/clamd

    START_CLAMD=yes

> vi /etc/clamav.conf

    # Example
    LogSyslog
    PidFile /var/run/clamd.pid

>   vi /etc/freshclam.conf

    #Example

> emerge qmail-scanner
> vi /var/qmail/control/conf-common

    SOFTLIMIT_OPTS="-m 16000000"

> vi /var/qmail/bin/qmail-scanner-queue.pl

    # Indien een van onderstaande vars leeg is moet je de scanner hercompileren
    my $clamscan_binary='/usr/bin/clamscan';
```

```
my $spamc_options=’ ’;
my $spamc_binary='/usr/bin/spamc';
```

> vi /var/qmail/control/conf-common

```
QMAILQUEUE="/var/qmail/bin/qmail-scanner-queue.pl" export QMAILQUEUE
```

```
> chown -R qmaild:nofiles /var/spool/qmailscan
> chown qmaild:nofiles /var/qmail/bin/qmail-scanner-queue.pl
```

```
> rc-update add spamd default
> rc-update add clamd default
> /etc/init.d/spamd start
> /etc/init.d/clamd start
```

- Spam koppelen aan MySQL database met user instellingen

> vi /etc/spamassassin/local.cf

```
# Toevoegen aan bestand
user_scores_dsn             DBI:mysql:spamassassin:localhost
user_scores_sql_username    dbusername
user_scores_sql_password    dbpassword
```

> mysql –u root -p

```
> create database spamassassin;
> use mysql;
> grant select, insert, update, delete, create, drop on spamassassin.* to
        spamassassin@localhost identified by 'abcd';
> flush privileges;
> use spamassassin
> CREATE TABLE userpref (
        username varchar(100) NOT NULL,
        preference varchar(30) NOT NULL,
        value varchar(100) NOT NULL,
        prefid int(11) NOT NULL auto_increment,
        PRIMARY KEY (prefid),
        INDEX (username)
      ) TYPE=MyISAM;
> quit;
```

```
> emerge dev-perl/Msql-Mysql-modules
> emerge dev-perl/DBI
> /etc/init.d/spamd restart
```

- Apache instellen en toevoegen aan runlevel

> vi /etc/conf.d/apache

APACHE2_OPTS="-D SSL -D PHP4"

> cd /usr/lib/apache2/conf/ssl
> openssl req -new -out server.csr

Fhjhretrkjf%63iu8dhjfhnb#@sFGHY
Country Name (2 letter code) [AU]:NL
State or Province Name (full name) [Some-State]:Noord-Holland
Locality Name (eg, city) []:Wormerveer
Organization Name (eg, company) [Internet Widgits Pty Ltd]:WireITup
Organizational Unit Name (eg, section) []:
Common Name (eg, YOUR name) []: hostnaam.wireitup.nl
Email Address []:j.barendse@wireitup.nl
haspaas

> openssl rsa -in privkey.pem -out server.key (vul bovenstaande wachtwoord in)
> openssl x509 -in server.csr -out server.crt -req -signkey server.key -days 365
> rm privkey.pem server.csr
> mkdir /var/log/apache2
> rc-update add apache2 default
> /etc/init.d/apache2 start

Enkele bronnen/handleidingen die gebruikt zijn voor deze handleiding:

Meer uitleg over email en spam op gentoo
http://gentoo-wiki.com/HOWTO_Setup_QMAIL_VPOPMAIL_and_Other_Mail_Servers

Programma om voorbeeld Spam config te maken
http://www.yrex.com/spam/spamconfig.php

VPOPMAIL en Horde infomatie
http://nav.bandersnatch.org/clues/vpopimp/vpopimp-hacks.html

#Spamassasin local.cf handleiding
http://www.iam.unibe.ch/~scg/Resources/SpamAssassin/Mail_SpamAssassin_Conf.html

#Handleiding voor het installeren van diverse daemons op Gentoo
http://www.gentoo.org/doc/en/qmail-howto.xml

Oplossing QMAIL SCANNER probleem
http://sourceforge.net/mailarchive/message.php?msg_id=9833605

Sites van de producten zelf
http://httpd.apache.org/, http://www.horde.org/, http://www.clamav.net/,
httpd://spamassasin.apache.org/, http://www.mysql.com/,
http://www.inter7.com/index.php?page=vpopmail, http://cr.yp.to/qmail.html,
http://www.courier-mta.org/imap/, http://www.gentoo.org/

Bronvermeldingen

[I] Open Standaarden Commissie, http://nllgg.kovoks.nl/nllgg_osc, NederLandse Linux Gebruikers Groep.

[II] Daniel A. Morgan , http://www.webservertalk.com/message884505.html, University of Washington, Januari 2005

[III] James Snell - Doug Tidwell – Pavel Kulchenko, Programming Web Services with SOAP – pagina 24, O'REILLY, ISBN 0-596-00095-2, Januari 2002

[IV] Standards and Technical Regulations Directorate, http://www.dti.gov.uk/strd/nssf.html, The Department of Trade UK

[V] Postel, J., "Internet Protocol" RFC 791, USC/Information Sciences Institute, September 1981.

[VI] Service Orientation and Its Role in Your Connected Systems Strategy, http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnbda/html/srorientwp.asp, Microsoft Corporation, Julie 2004

[VII] Thomas Erl, afbeelding 3.3 , http://www.informit.com/articles/article.asp?p=336265&seqNum=1, Informit.com, 3 september 2004

[VIII] Thomas Erl, afbeelding 3.4 , http://www.informit.com/articles/article.asp?p=336265&seqNum=1, Informit.com, 3 september 2004

[IX] Gentoo, http://www.gentoo.org/main/en/about.xml

[X] D.J. Bernstein, qmail: the Internet's MTA of choice , http://cr.yp.to/qmail.html

[XI] Vpopmail, http://www.inter7.com/index.php?page=vpopmail, Inter7 Internet Technologies
[XII] MySQL Database, http://www.mysql.com/, MySQL AB

[XIII] Courier IMAP, http://www.courier-mta.org/imap/, Courier MTA

[XIV] SpamAssassin, http://wiki.apache.org/spamassassin/SpamAssassin, Apache Software Foundation

[XV] Clam Antivirus, http://www.clamav.net/abstract.html

[XVI] Horde IMP, http://www.horde.org/imp/, Horde Project

[XVII] HTTPD, http://httpd.apache.org/ , Apache Software Foundation

[XVIII] M. Wahl - T. Howes - S. Kille, Lightweight Directory Access Protocol , http://www.ietf.org/rfc/rfc2251.txt , Network Working Group RFC, december 1997

[XIX] J. Case - M. Fedor - M. Schoffstall - J. Davin, Simple Network Management Protocol, http://www.faqs.org/rfcs/rfc1157.html, Network Working Group RFC, Mei 1990

[XX] M. Rose - K. McCloghrie - Concise MIB Definitions, http://www.cse.ohio-state.edu/cgi-bin/rfc/rfc1212.html, Network Working Group RFC, maart 1991

[XXI] James Snell - Doug Tidwell – Pavel Kulchenko, Programming Web Services with SOAP – Hoofdstuk 1 t/m 5, O'REILLY, ISBN 0-596-00095-2, Januari 2002

[XXII] Scott Nichol, Introduction to NuSOAP, http://www.scottnichol.com/nusoapintro.htm, 3 november 2004

[XXIII] Paul Kulchenko, Quick Start with SOAP, http://www.perl.com/pub/a/2001/01/soap.html, 29 janurie 2001

[XXIV] Robert A. van Engelen, Gsoap, http://www.cs.fsu.edu/~engelen/soap.html