Onderzoek naar een professionele ICT infrastructuur

Andree Toonk
Leendert van Doesburg

2 februari 2004

Samenvatting

In de huidige maatschappij worden ICT diensten steeds belangrijker. Een trend is dat diverse ICT diensten de laatste jaren steeds meer gecentraliseerd worden. Een gevolg van dit centraliseren is dat de diensten met veel zorg opgezet en onderhouden dienen te worden. Bij het opzetten van een ICT dienst is het belangrijk rekening te houden met een aantal functionele en niet functionele eisen. Met name de email infrastructuur dient met veel zorg opgezet te worden, omdat vooral deze dienst erg hard groeit. Schaalbaarheid is dus een eis welke voor deze dienst belangrijk is. Door het scheiden van inkomende en uitgaande smtp flows en het opdelen van de mail infrastructuur in verschillende lagen (tiers) kan dit worden bereikt. Voor de webservices gelden dezelfde eisen, niet alleen schaalbaarheid, maar zeker ook de performance, beschikbaarheid en security spelen hierbij een beangrijke rol. Juist doordat steeds meer aankopen via het world wide web worden gedaan, is het voor de inkomsten van een organisatie belangrijk dat deze dienst een zo hoog mogelijke beschikbaarheid heeft. Om dit te bereiken zijn verschillende mogelijkheden denkbaar. Centrale authenticatie en authoristatie spelen een steeds grotere rol binnen de huidige organisaties. Dit kan worden gerealiseerd door middel van LDAP. LDAP wordt niet langer alleen maar gebruikt voor authenticatie en authorisatie, maar ook voor het bij houden van een digitale agenda en een persoonlijke adressen boek. Het is dus belangrijk de LDAP infrastructuur degelijk op te zetten. Doordat LDAP een steeds meer centrale rol krijgt, spelen met name de security en de beschikbaarheid en belangrijke rol. Traditioneel had iedere dienst zijn eigen lokale storage. Technieken zoals NAS en SAN bieden een gecentraliseerd alternatief. Met name SAN technieken hebben de laatste jaren veel aan terrein gewonnen. Het opzetten van een ICT infrastructuur is stap één, de tweede en minstens zo belangrijke stap is het beheren van de diensten. Een provisioning systeem kan het beheer vereenvoudigen, zodat dit efficiënter uitgevoerd kan worden. Door de diensten pro-aktief te beheren kunnen problemen vroegtijdig worden herkend en worden bestreden. Om pro-aktief te kunnen beheren is monitoring belangrijk, door de metingen van de monitoring de registreren en uit te zetten in grafieken zijn trends eenvoudig te herkennen.

Voorwoord

Dit document is het resultaat van het Analytisch Server Project welke door ons aan de opleiding Systeem- en Netwerkbeheer van de UvA wordt uitgevoerd. Dit project liep van 5 jan t/m 30 jan 2004. Er is door ons een onderzoek gedaan naar hoe een professionele ICT-infrastructuur wordt opgezet.

Het project is deels uitgevoerd bij het bedrijf ZXFactory te Ede. ZXFactory is gespecialiseerd in het beheren en faciliteren van complexe ICT-omgevingen. Voorbeelden hiervan zijn het beheer van de servers van Nltree (kennisnet), Multikabel en Casema. ZXFactory heeft in de loop der jaren veel ervaring opgedaan met het beheren en het opzetten van serverparken. Dit bedrijf bood ons de mogelijkheid om inzicht te krijgen in de door hun beheerde omgevingen. Hun gastvrijheid en open opstelling hebben wij zeer op prijs gesteld. In het bijzonder gaat onze dank uit naar Jeroen de Meijer welke ons met grote enthousiasme in raad en daad heeft bijgestaan.

Dit rapport is geschreven in de vorm van een consultancyrapport. Het maken van dit rapport heeft ons inzicht gegeven in hoe een grote ICT-omgeving opgezet kan worden met de bijbehorende aspecten. We hopen dat de lezer na het lezen van dit rapport hiervan eveneens een duidelijk beeld heeft gekregen.

Andree Toonk
Leendert van Doesburg
Ede, januari 2004

Inleiding

ICT-infrastructuren worden vandaag de dag steeds belangrijker. Een trend is dat diensten binnen een organisatie steeds meer gecentraliseerd worden. Dit heeft een andere aanpak en opzet van de diensten tot gevolg. In dit rapport wordt uiteen gezet hoe een dergelijke professionele omgeving opgezet kan worden. Hiervoor is een onderzoek uitgevoerd waarbij bestaande ICT-infrastructuren als referentie zijn gebruikt. De centrale vraag binnen dit onderzoek was: "Hoe wordt een professionele ICT-infrastructuur opgezet?"Het resultaat hiervan is in dit rapport beschreven. Hierbij is getracht in conceptuele vorm te werken. Vendor specifieke implementaties en oplossingen worden buiten beschouwing gelaten.

Bij het opzetten van een grote ICT-infrastructuur dient er aan een aantal eisen voldaan te worden. Deze eisen worden toegelicht in hoofdstuk 1. Een ICT-infrastructuur is het makkelijkst te beschrijven als deze wordt opgedeeld in verschillende lagen. In dit rapport wordt een drie tiermodel gehanteerd. Welke tiers dit zijn en de functie hiervan, worden beschreven in hoofdstuk 1. In de hoofdstukken daarna worden veel gebruikte diensten zoals WWW, SMTP, POP en IMAP volgens het 3-tier model behandeld. Per tier wordt gekeken naar de in de hoofdstuk 1 genoemde eisen. Ieder hoofdstuk wordt vervolgens afgesloten met een voorbeeld architectuur.

Dataopslag speelt binnen de huidige ICT-infrastructuur een belangrijke rol en zal daarom afzonderlijk behandeld worden. Verschillen technologieën zoals SAN en NAS zullen daarbij aan bod komen. Verder worden naast security de beheerspecifieke onderwerpen monitoring en provisioning in afzonderlijke hoofdstukken behandeld.

Met dit rapport wordt getracht de lezer een beeld te geven hoe een professionele ICT-infrastructuur kan worden opgezet.

Inhoudsopgave

Hoofdstuk 1

Architectuur

Aan het opzetten van een ICT-infrastructuur worden bepaalde eisen gesteld. Dit zijn zowel functionele als niet-functionele eisen welke kunnen variëren van "time to market"tot "performance". In dit rapport wordt ingegaan op de technische eisen van aangeboden services binnen grote ICT-infrastructuren. De services zullen op server-niveau beschreven worden.

Het ontwerp van een architectuur van een infrastructuur wordt bekeken aan de hand van de volgende eisen:

- Scalability
- Performance
- Availability en Reliability
- Security
- Manageability
- Adaptability

1.1 Scalability

Een infrastructuur wordt ontworpen voor een bepaald aantal gebruikers. Dit aantal zal in de toekomst in veel situaties toenemen. Ook is het mogelijk dat huidige gebruikers intensiever gebruik gaan maken van bepaalde services. Het moet mogelijk zijn om de infrastructuur op deze veranderingen aan te passen. Hoe eenvoudig het is om na het initiële ontwerp het aantal te faciliteren gebruikers uit te breiden zonder de architectuur opnieuw te ontwerpen bepaald de mate van schaalbaarheid. Er is onderscheid te maken tussen twee schalingstechnieken: verticale en horizontale schaling. Verticale schaling wordt gerealiseerd door servers van extra resources zoals CPU, geheugen en I/O te voorzien. Horizontale schaling is mogelijk door op systeemniveau meer servers met dezelfde functie parallel te plaatsen.

1.2 Performance

De performance wordt bepaald door de response tijd. Dit is de vertraging waarna een service reageert op een verzoek van de gebruiker of een andere service. Performance hangt samen met de schaalbaarheid. Door teveel gebruikers van een bepaalde server gebruik te laten maken, zal deze server veel verzoeken per tijdseenheid moeten verwerken en kan de response tijd oplopen. De server wordt dus zwaarder belast. Door horizontaal of verticaal te schalen, wordt deze belasting verlaagd. De performance is in veel gevallen maatgevend voor het bepalen of een infrastructuur uitgebreid moet worden.

1.3 Availability en Reliability

Door de beschikbaarheid van de services te verhogen neemt de betrouwbaarheid toe. Vooral als een service in een infrastructuur gecentraliseerd wordt aangeboden, maken relatief veel gebruikers hiervan gebruik. De service is een single points of failure (SPOF). Door redundantie in te bouwen, neemt de beschikbaarheid en de betrouwbaarheid toe. Mogelijkheden hiervoor zijn clustering van servers en het dubbel uitvoeren van hardware. De upime van de services bepaald de beschikbaarheid. Deze kan worden uitgedrukt in procenten. Gehanteerde variabelen hierbij zijn de mean time between failure (MTBF) en de mean time to repair (MTTR).

$$Availability = \frac{MTBF}{MTBF + MTTR}$$

Uptime Percentage	*Nines*	*Allowable Downtime Per Month*
99.9999	6	0.043 minute
99.999	5	0.43 minute
99.99	4	4.30 minutes
99.9	3	43 minutes
99	2	7.2 hours

Tabel 1.1: *Uptime absoluut / relatief*

Tabel 1.1 geeft de MTBF weer, hier is absoluut tegen relatief uitgezet[1]

1.4 Manageability

Hoe eenvoudiger de architectuur is opgezet, hoe eenvoudiger deze is te beheren (KISS, keep it simple stupid). Een complexe infrastructuur verhoogt de kans op menselijke fouten. Verder zijn personele kosten hoog. Een eenvoudige infrastructuur vereist minder beheerders. Een grote infrastructuur met gecentraliseerd beheer vraagt om schaalbare beheerstechnieken. Het beheer dient zo mogelijk tot het minimum beperkt te worden. Automatiseren van veel voorkomende beheersprocessen kan vooral bij grote infrastructuren efficiënt zijn. Management tools, provisioning en monitoring zijn middelen welke het beheer eenvoudiger maken.

[1] Priscilla Oppenheimer, Top-Down Network Design, Cisco Press, 1999.

1.5 Adaptability

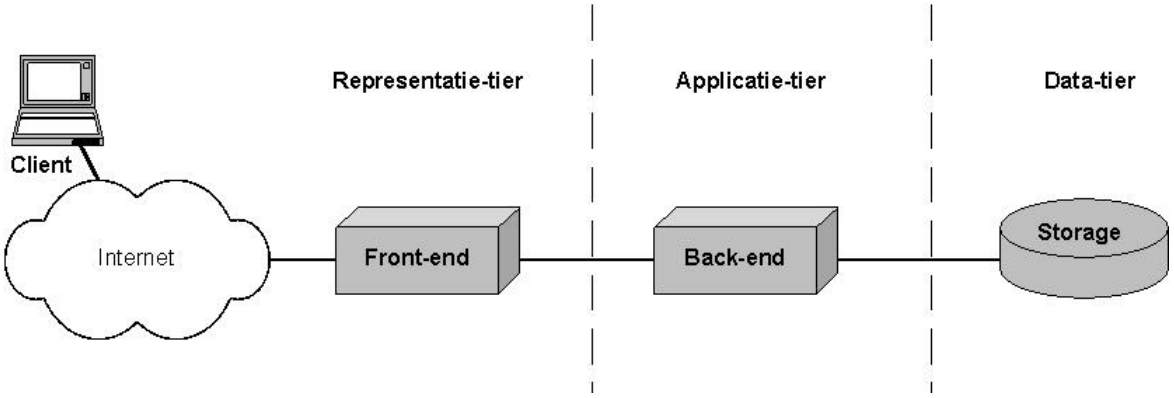
Adaptability bepaald de mate van flexibiliteit waarmee een infrastructuur aan te passen is. In iedere infrastructuur zullen in de loop der tijd wijzigingen worden gemaakt om aan zekere wensen te voldoen. De adaptability is te verhogen door gebruik te maken van open standaarden. Door de architectuur overzichtelijk en modulair op te bouwen is het mogelijk om delen van het ontwerp eenvoudig te vervangen.

1.6 Security

Security is een belangrijk en complex onderdeel van de iedere ICT-infrastructuur. De rol hiervan wordt in veel organisaties onderschat. Security is in breder perspectief een combinatie van processen, producten en mensen. Het beveiligen van services beperkt zich dus niet tot de infrastructuur maar gaat breder. Het verhogen van de security kan bereikt worden door security-policies binnen een organisatie te hanteren. Binnen de infrastructuur kan security op de netwerklaag en op de systeemlaag worden aangepakt. Er zal vooral ingegaan worden op het beveiligen op de systeemlaag.

1.7 Tier model

De architectuur van een grote infrastuctuur kan in meerdere tiers worden verdeeld. Met de modulaire opbouw wordt flexibiliteit gecreëerd. Het modulair opzetten van een infrastructuur maakt het mogelijk om de bovengenoemde zes eisen makkelijker te implementeren. Individuele componenten kunnen zo afzonderlijk worden beschreven. Modules kunnen makkelijker toegevoegd worden. Het maakt de architectuur overzichtelijk.



Figuur 1.1: *3-tier model*

De verschillende architecturen worden in dit onderzoek bekeken en beschreven aan de hand van een zelf gedefinieerd 3-tier model[2] (zie figuur 1.1). Deze drie tiers zijn:

[2]Het kan zijn dat dit model niet overeen komt met 3-tier modellen zoals deze in andere documenten gehanteerd worden.

- Eerste tier (representatie-tier)
- Tweede tier (applicatie tier)
- Derde tier (data-tier)

1.7.1 Eerste tier

De representatie-tier is de tier waar de gebruikers mee communiceren. Deze tier verzorgt de presentatie van de services naar de gebruikers toe. Gebruikers zien alleen deze tier, achterliggende tiers zijn voor hen onzichtbaar. De complexiteit van de infrastructuur wordt met deze tier voor de gebruiker afgeschermd. Deze tier bevat de front-end servers welke typisch horizontaal geschaald worden.

1.7.2 Tweede tier

De applicatie-tier bevat de applicatie-servers van de infrastructuur. Deze servers zijn alleen vanaf de eerste tier en niet rechtstreeks voor de gebruikers toegankelijk. De tweede tier bevat de back-end servers van de aangeboden services. Op deze tier wordt vaak verticaal geschaald.

1.7.3 Derde tier

De data-tier bevat de dataopslag van de infrastructuur. Verschillende oplossingen zoals SAN en NAS horen op deze tier thuis. Deze tier bevat de verschillende opslagformaten en de wijze waarop deze beschikbaar gesteld worden.

Hoofdstuk 2

SMTP

Simpel Mail Tranfer Protocol (SMTP) is verantwoordelijk voor het versturen en het routeren van email. Dit protocol speelt een belangrijke rol in de huidige mailinfrastuctuur. De servers waarop een SMTP server draait worden ook wel Mail Transfer Agents (MTA) genoemd. Het totale volume aan email berichten blijft in een stijgende lijn toenemen, wat vooral te wijten is aan de toename van spam en virus berichten. Dit kan tot gevolg hebben dat er bij de MTA bottlenecks kunnen ontstaan wat resulteert in een lagere availability van de maildiensten. Het gevolg hiervan is dat emailberichten met vertraging aankomen.

Het is dus belangrijk goed na te denken over hoe de email architectuur wordt opgezet. Eén aspect hieruit zijn de MTA's. Voorbeelden van op SMTP gebaseerde MTA's zijn:

- Sendmail
- Postfix
- Qmail
- Iplanet/SunOne
- Openwave

De specificaties van het smtp protocol zijn gedefinieerd in RFC 2821: "Simple Mail Transfer Protocol". Enkele andere gerelateerde RFC's zijn:

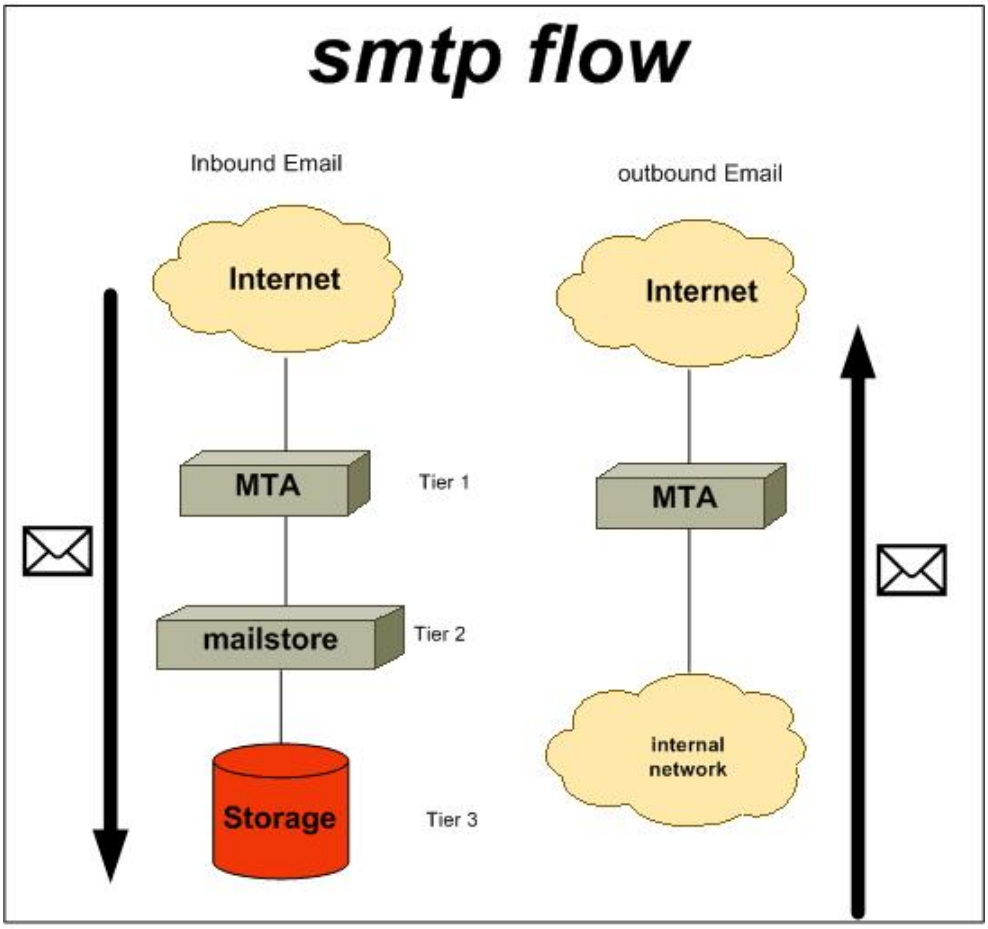
- RFC 2822: "Internet Message Format"
- RFC 2920: "SMTP Service Extension for Command Pipelining"
- RFC 3030: "SMTP Service Extensions for Transmission of Large and Binary MIME Messages"
- RFC 2847: "SMTP Service Extension for Secure SMTP over TLS"

Eén van de kenmerken van een goed emailserver is dat deze RFC compliant is, zodat deze zonder problemen kan communiceren met andere mailservers. Andere kenmerken waaraan bij het opzetten van een email infrastructuur moet worden gedacht zijn:

- Scalability

- Performance
- Availability en Reliability
- Security
- Manageability
- Adaptability

Bovenstaande punten zijn het beste te realiseren door de email infrastructuur modulair op te zetten. Een manier om dit te realiseren is door deze op te delen in drie lagen. Het 3 tier model voor SMTP ziet er uit als in figuur 2.1.



Figuur 2.1: *smtp-flow*

Zoals duidelijk te zien is in figuur 2.1, bestaan er separate flows voor inkomende en uitgaande email. De MTA voor uitgaande email wordt door gebruikers binnen het eigen netwerk gebruikt om mail te versturen, deze wordt in de voorbeelden smtp.os3.nl genoemd. Voor de inkomende email flow is

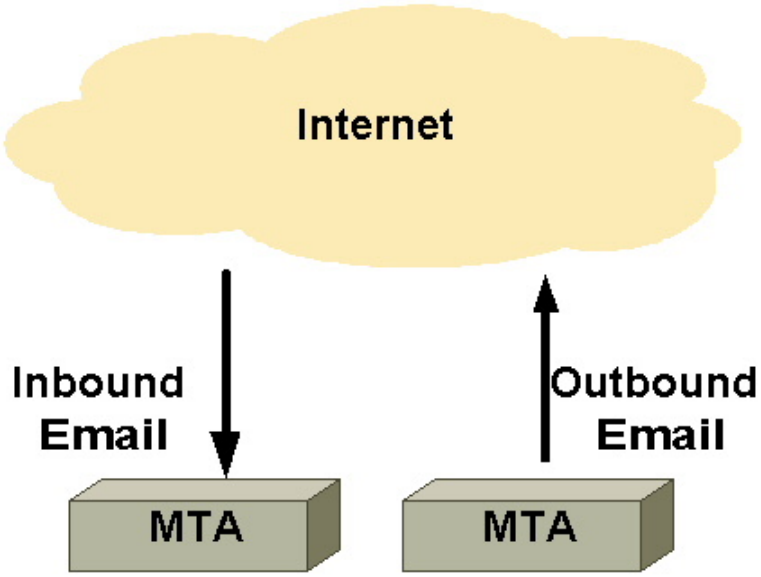
ook een aparte MTA beschikbaar. Dit zijn de mx server(s) welke in het voorbeeld mx.os3.nl wordt genoemd. De inbound email zal een langere weg binnen het eigen netwerk afleggen (zie figuur 2.1). Nadat de inbound email langs de eerste MTA is gekomen, zal deze worden door gestuurd naar de eigenlijke mailserver welke ook toegang heeft tot de mailbox van de gebruiker (mailstore).

2.1 Tier 1

De eerste tier is de representatie of access laag. Hierin staan de front-end systemen waarmee de gebruikers/clients communiceren. In het geval van SMTP komen hier de emailberichten binnen.

2.1.1 Scalability

Binnen deze eerste tier is het mogelijke een verdere scheiding aan te brengen. Om de schaalbaarheid te vergroten is het verstandig om de inkomende en uitgaande smtp flows te scheiden. Op deze manier kunnen eventuele problemen geïsoleerd worden tot alleen de uitgaande of de inkomende smtp flow. Een voorbeeld architectuur zou er alsvolgt uit kunnen zien, zie figuur 2.2.



Figuur 2.2: *gescheiden MTA flow*

Op deze manier kunnen ook aparte services zoals spam en virus scanning eenvoudig worden geïntegreerd op één van de flows. Door het scheiden van de flows hebben de extra services geen nadelige invloeden op de andere flow. Schaalbaarheid op dit niveau wordt vooral bereikt door "horizontaal"te schalen. Doordat deze tier stateless is, kunnen extra systemen eenvoudig bijgeplaatst worden als dit nodig is.

2.1.2 Availability en Reliability

Om de availability en reliability te verhogen kunnen extra systemen worden toegevoegd. Hiervoor zijn meerdere mogelijkheden. Zo kan er gebruik gemaakt worden van roundrobin DNS of loadbalancers. Aan de hand van een voorbeeld wordt round robin DNS gekeken. Als uitgaande mailserver binnen een organisatie gebruiken gebruikers smtp.os3.nl. In de DNS is smtp.os3.nl gedefinieerd met meerdere A records:

```
smtp.os3.nl.  IN A 172.16.42.25
smtp.os3.nl.  IN A 172.16.42.26
```

Nu wordt het verkeer naar smtp.os3.nl door middel van round robin dns verdeeld over beide (outbound) smtp server, namelijk 172.16.42.25 en 172.16.42.26.

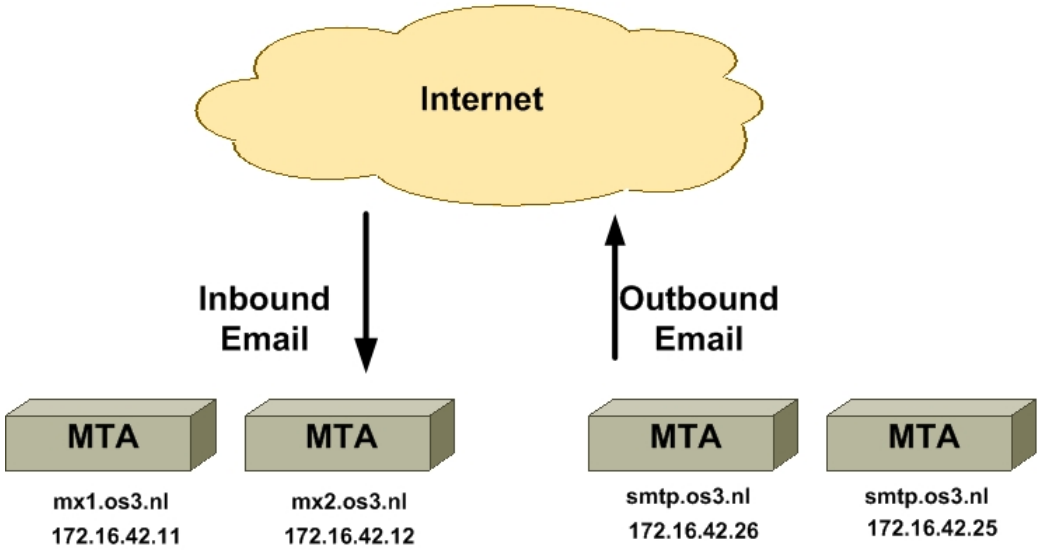
Voor de Inbound email wordt hetzelfde gedaan. De mx records voor het domein os3.nl zijn als volgt gedefinieerd:

```
os3.nl.             IN MX  10 mx1.os3.nl.
os3.nl.             IN MX  10 mx2.os3.nl.
os3.nl.             IN MX  20 backup-mx.os3.nl.
mx1.os3.nl.         IN A    172.16.42.11
mx2.os3.nl.         IN A    172.16.42.12
backup-mx.os3.nl.   IN A    10.16.42.13
```

Voor het domein os3.nl zijn een drietal mx records gedefinieerd; namelijk de mx1.os3.nl, mx2.os3.nl en backup-mx.os3.nl. De mx1.os3.nl en mx2.os3.nl hebben beide een priority van 10. Mochten deze allebei niet bereikbaar zijn, dan is er nog een derde mx host voor dit domein. Deze heeft een prioriteit van 20, wat betekent dat deze alleen wordt gebruikt wanneer de mx1 en mx2 niet bereikbaar zijn. De kans dat zowel de mx1 en mx2 niet bereikbaar zijn is klein, maar zou bijvoorbeeld kunnen voorkomen wanneer er netwerk/routerings probleem zijn. De mx host backup-mx.os3.nl dient daarom bij voorkeur in een ander netwerk te worden geplaatst. Deze backup mx host fungeert als een grote tijdelijke queue. Wanneer de eigenlijk mx servers down zijn of niet bereikbaar, wordt de mail afgeleverd bij backup-mx.os3.nl. Deze bewaart de mail in een queue totdat de mail weer afgeleverd kan worden bij de mx1.os3.nl of mx2.os3.nl. Schematisch ziet dit eruit als in figuur 2.3.

Hoewel round robin dns een goedkope manier is om load balancening te realiseren, is het echter niet altijd de meest effectieve manier. Hoewel zowel de availability en reliability hiermee wel verhoogd worden, is deze zeker niet gegarandeerd. Wanneer in het bovenstaande voorbeeld mx1.os3.nl niet langer beschikbaar is, zullen er toch altijd nog mailservers zijn welke bij mx1.os3.nl proberen mail af te leveren. Dit komt omdat deze als mx host in de DNS server staat. Round robin dns is dus statisch en kan niet, of relatief langzaam, reageren op outages van een mx host. Om een hogere availability en reliability te bereiken kan beter gewerkt worden met loadbalancers. Loadbalancers hebben over het algemeen meer kennis van de te loadbalancen dienst. Deze voeren ook checks uit op de service en wanneer een check niet slaagt, zal de betreffende server uit de "loadbalancing group"gehaald worden en dus niet langer gebruikt worden. In het voorbeeld van os3.nl. worden de dns entries nu alsvolgt gedefinieerd:

```
smtp.os3.nl.        IN A 172.16.42.25
```



Figuur 2.3: *mta-roundrobin architectuur*

```
os3.nl.              IN MX  10 mx1.os3.nl.
os3.nl.              IN MX  20 backup-mx.os3.nl.
mx1.os3.nl.          IN A 172.16.42.11
backup-mx.os3.nl.    IN A 10.16.42.13
```

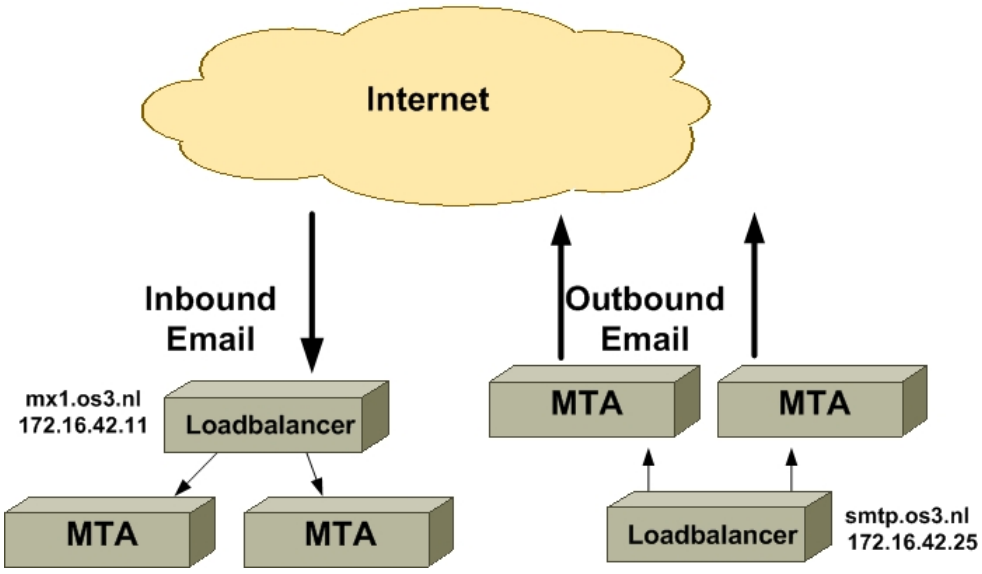
In het voorbeeld bestaan er geen dubbele dns entries (dns round robin), maar heeft elke dienst slechts één IP-adres. Dit IP-adres is geconfigureerd op de loadbalancer, welke de verzoeken over de servers verdeelt. Schematisch ziet dit eruit als in figuur 2.4.

In figuur 2.4 is de loadbalancer echter een single point of failure. Het is dan ook aan te raden deze redundant uit te voeren. Dit is mogelijk met technieken als VSRP en HSRP. In deze architectuur is het nog wel steeds aan te raden een backup mx (een secondaire mx host met een hogere priority) in een ander netwerk te plaatsen.

2.1.3 Performance

Hoewel deze eerste tier relatief goed schaalbaar is door extra systemen bij te plaatsen, zijn er natuurlijk ook mogelijkheden om de performance van een individueel systeem zo hoog mogelijk te krijgen. Op dit niveau worden performance problemen typisch veroorzaakt door een relatief langzame verwerking van de mailqueue. Wanneer een email binnenkomt, wordt bekeken wat hiermee gedaan moet worden.

- Bestaat het doel adres?
- Waar moet mail voor deze gebruiker naar toe worden gestuurd?
- Vacation messages?
- Alias?



Figuur 2.4: *mta-loadbalance architectuur*

- sanity checks

De meeste mailservers werken alsvolgt; email komt binnen en wordt in de queue geplaatst. Dan wordt er gewacht totdat deze verwerkt kan worden. De software welke dit doet haalt de mail op uit de queue en verwerkt deze verder. Deze handelingen kunnen allemaal snel worden verricht. De bottleneck treedt vaak op bij het schrijven naar de mailqueue. De mailqueue bevindt zich op de storage van de mta. Harddisks zijn relatief langzaam wanneer je deze vergelijkt met de andere componenten binnen een systeem. Iedere keer wanneer een systeem een email in de queue plaats moet deze wachten op een "commit"van het OS. Een commit is de bevestiging van het OS dat de data is weg geschreven naar de schijf. Wanneer een mailserver een grote hoeveelheid emails te verwerken krijgt, is dit typische de plek waarbij de bottleneck ontstaat. Een oplossing voor dit I/O probleem is om gebruik te maken van de cache welke op veel harde schijven aanwezig is. Het schrijven naar de cache gaat vele malen sneller dan naar de harddisk zelf. Over het algemeen zullen de berichten vaak ook niet echt op de harddisk weg geschreven worden omdat een ander proces ze voor die tijd vaak alweer uit de mailqueue en dus de cache ophaalt om verder te verwerken. Een normale harddisk heeft gemiddeld een paar MB aan cache. Dit is over het algemeen te weinig voor een beetje mailqueue. Een mogelijke oplossing is om gebruik te maken van een storage array. Deze bevatten meestal enkele honderden megabytes tot zelfs een gigabyte cache on board. Op deze manier wordt de performance van een mailsysteem verbeterd. Het uiteindelijke resultaat zal zijn dat er meer emails per seconde verwerkt kunnen worden. Wel dient hierbij nog opgemerkt te worden dat het belangrijk is de harddisk of storage array met een backup batterij uit te rusten. Omdat de cache vluchtig geheugen is zal in het geval van een power outage de data in de cache verloren gaan waardoor emailberichten verdwijnen. Het gebruik van een extra backup voeding of batterij elimineert dit potentiële probleem. De performance is verder te verhogen door voldoende cpu power en RAM geheugen te gebruiken.

2.1.4 security

De MTA's zijn rechtstreeks verbonden met Internet en dienen daarom goed beveiligt te zijn. Bij voorkeur dient de server alleen op poort 25 (tcp) te benaderen te zijn. Uiteraard is het ook belangrijk dat de software up to date blijft. Het is dus belangrijk om de patches op tijd te installeren. Dit kan één van de redenen zijn om niet te kiezen voor een bepaald product. Zo staat sendmail bekend om de security problemen. Dit wil niet zeggen dat men niet voor sendmail moet kiezen, maar er dient wel rekening mee gehouden worden. Een ander security aspect is de configuratie van de MTA. Wanneer een mailserver verkeerd geconfigureerd is, kan deze meer informatie over het systeem of de mail infrastructuur weergeven dan gewenst is. De meeste van deze commando's zijn verwerkt in het ESMTP protocol. Het kan een keuze zijn dit protocol niet te gebruiken en alleen gebruik te maken van SMTP. Het is in de meeste mailserver configuraties ook mogelijk enkele van deze opties uit te schakelen. Over de volgende opties moet nagedacht worden bij het configureren van een MTA:

- VERIFY (VRFY)
- EXPAND (EXPN)

Deze commando's kunnen potentiëel teveel informatie over de organisatie weergeven. VERIFY wordt gebruikt om te controleren of een user lokaal bekent is, en wat eventueel zijn echte naam en forward adres is. Deze functie is te vergelijken met "finger". Zie onderstaande voorbeelden (s=send, r=receive):

```
   S: VRFY Smith
   R: 250 Fred Smith <Smith@USC-ISIF.ARPA>
Or
   S: VRFY Smith
   R: 251 User not local; will forward to <Smith@USC-ISIQ.ARPA>
Or
   S: VRFY Smith
   R: 550 String does not match anything.
Or
   S: VRFY Smith
   R: 551 User not local; please try < Smith@USC-ISIQ.ARPA>
Or
   S: VRFY Smith
   R: 553 User ambiguous.
```

Zoals is te zien geeft dit redelijk wat informatie over de user Smith. Dit hoeft niet per definitie slecht te zijn, maar er dient wel over nagedacht worden. Dit zelfde geld voor de EXPAND optie:

```
   S: EXPN Example-People
   R: 250-Jon Postel <Postel@USC-ISIF.ARPA>
   R: 250-Fred Fonebone <Fonebone@USC-ISIQ.ARPA>
   R: 250-Sam Q. Smith <SQSmith@USC-ISIQ.ARPA>
   R: 250-Quincy Smith <@USC-ISIF.ARPA:Q-Smith@ISI-VAXA.ARPA>
   R: 250-<joe@foo-unix.ARPA>
   R: 250 <xyz@bar-unix.ARPA>
```

Met EXPAND optie kunnen alias of mailinglist adressen worden "uitgeklapt". In het bovenstaande voorbeeld wordt dit gedaan voor de list "Example-People". Ook voor dit commando geldt, dat dit niet per definitie slecht of onveilig is. Maar men dient hier in ieder geval over nadenken, zodat niet ongewenst teveel gegevens op te vragen zijn.

De minimum opties welke iedere MTA in ieder geval dient te ondersteunen zijn:[1]

- HELO
- MAIL
- RCPT
- DATA
- RSET
- NOOP
- QUIT

Andere opties om het SMTP verkeer te beveiligen zijn het gebruik van SSL/TLS (STARTTLS) en het eerst authoriseren voor dat mail verstuurd mag worden (AUTH).

2.1.5 Manageability

Afhankelijk van welk MTA product er gebruikt wordt, zijn er verschillende manieren waarop de MTA software te configureren is. Meestal is er de mogelijkheid dit te doen door middel van één of meerdere configuratie files. Bij sommige pakketen is er de mogelijkheid dit via een GUI te doen. Een ander alternatief is de configuratie te halen uit een directory server. Er zijn een aantal algemene parameters welke voor ieder MTA product geconfigureerd dienen te worden. Eén van de belangrijkste parameters is de wijze waarop de MTA de user gegevens kan achterhalen. Dit kan gedaan worden door middel van een lokale (password) file een database of LDAP. In grote organisaties wordt vaak gebruik gemaakt van een gecentraliseerde database. Een LDAP server is in feite ook een database, maar met enkele specifieke eigenschappen (zie hoofdstuk LDAP). Wanneer een MTA een email bericht binnen krijgt moet deze aan de hand van het "mail to" adres enkele parameters opzoeken:

- Moet de MTA mail afhandelen voor dit domein?
- Bestaat deze user?
- Op welke mailstore zit deze user?

Al deze gegevens kunnen worden verkregen uit een database, lokale file of LDAP server. Vanuit beheers oogpunt kan het voordelen hebben om de configuratie centraal op te slaan. Zodat wanneer er gebruik wordt gemaakt van meerdere MTA's (met dezelfde configuratie) de wijzigingen maar één keer doorgevoerd hoeven te worden. Het gebruik van een LDAP server is ten zeerste aan te raden, omdat niet alleen de MTA maar ook diverse andere services account gegevens nodig hebben.

[1] RFC 2821 section 4.5.1 commands

Een lokale file met deze gegevens (passwd) zou niet schalen. Een frontend kan uit de loadbalancing group gehaald worden en er kan onderhoud op gepleegd worden, of eventueel zelfs gereboot worden zonder dat de gebruikers hier iets van merken. Wijzigingen aanbrengen in de configuratie of patches installeren op één van de frontend MTA's is door het gebruik van loadbalancers relatief eenvoudig.

2.1.6 Adaptability

Door dat de architectuur in verschillende tiers is onderverdeeld, is deze eenvoudiger aan te passen. Op de eerste tier is het relatief eenvoudig om bijvoorbeeld van product te wisselen. Doordat de frontend MTA geen vendor specifieke dingen hoeven te doen is het eenvoudig een frontend MTA van vendor Y te vervangen door die van vendor Z. Dit gaat zowel op voor inbound als outbound email.

2.2 Tier 2

In de tweede tier vinden we de eigenlijke mailserver applicatie. In dit geval is dat de MTA, maar in de praktijk zal deze mailserver applicatie ook vaak de pop3/imap4 server zijn. Dit is de server welke toegang heeft tot de mailboxen van de gebruikers, vanaf nu zal deze mailserver de mailstore genoemd worden. Deze tweede tier wordt alleen gebruikt voor inbound email.

2.2.1 Scalability

Het is op dit niveau niet altijd mogelijk om horizontaal te schalen. Het bij plaatsen van een tweede parallelle applicatie brengt namelijk een aantal problemen met zich mee, zoals de toegang tot de mailbox. Schaalbaarheid op dit niveau wordt bereikt door voornamelijk in de hardware te schalen. Dit betekent een groot chasis met voldoende uitbreidings mogelijkheden voor zowel memory als CPU en Disk. Ook door gebruik te maken van clustering kan de schaalbaarheid worden vergroot (zie ook Availability en Reliability)

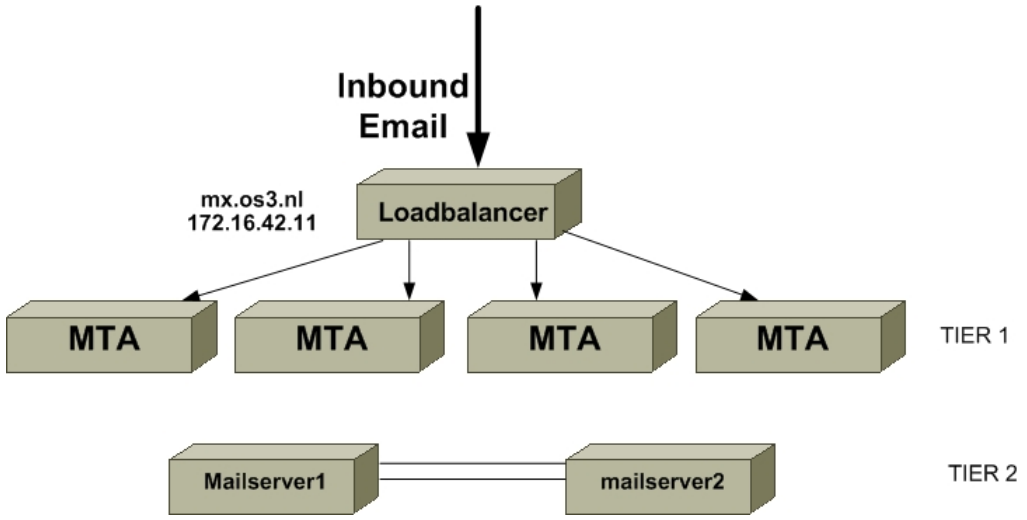
2.2.2 Performance

Om de performance van de applicatie tier te verhogen is het een goed idee om de users te scheiden over twee of meerdere mailstores. Op welke mailstore een user gedefinieerd is, staat vermeld in de LDAP of database server. Op deze manier kan de load worden verdeeld over twee of meerdere servers, dit komt ten goede aan de performance.

2.2.3 Availability en Reliability

Op de tweede tier zijn er geen meerdere servers die dezelfde taken vervullen. Hierdoor zijn availability en reliability een belangrijke en ingewikkelde zaak. Op tier 1 was het mogelijk meerdere servers parallel te zetten achter een load balancer. Dit kan omdat dat deze stateless zijn. De tweede tier is echter niet langer stateless. Users zijn bekend op één van deze twee mailserver applicaties (in het geval van twee mailstores). Om toch de Availability en Reliability te verhogen is het mogelijk de mailservers in een cluster onder te brengen. In een normale situatie zijn beide mailservers actief, deze beide nodes checken elkaars availability. In het geval van een outage van één van de twee nodes

zal de andere node de functie van de eerste node overnemen. Op dat moment draaien beide mailserver applicaties dus op één node. Dit heeft tot gevolg dat in het geval van een outage van één van de nodes de performance omlaag zal gaan, echter de availability blijft gegarandeerd. Voorwaarde van deze architectuur is wel dat de servers bij elkaars storage kunnen komen, zie hiervoor de paragraaf over tier3.



Figuur 2.5: *MTA in cluster opstelling*

2.2.4 Security

De mailservers in de tweede tier dienen nooit direct benaderbaar te zijn vanaf Internet, dit moet altijd gaan via één van de MTA frontends. Dit is voor het security aspect van de servers positief. Theoretisch gezien kan men er voor kiezen bepaalde patches niet of op een later moment te installeren. Dit is uiteraard afhankelijk van de advisory behorende bij de patch.

2.2.5 Manageability

Het beheer van de mail servers is door het gebruik van een cluster een stuk ingewikkelder geworden. Om de applicatie en het OS goed te kunnen beheren is ook kennis nodig van de cluster software. Het is niet mogelijk om "zomaar" even een node down te brengen, omdat dit performance problemen met zich kan mee brengen. Om de availability en reliability te verhogen kan gebruik gemaakt worden van een cluster. Dit brengt echter complexiteit met zich mee wat de manageability niet ten goede komt. Hier moet dus een afweging tussen worden gemaakt.

2.2.6 Adaptability

Wijzigingen in de architectuur op dit niveau zullen minder snel gemaakt worden dan die op de eerste tier. Dit heeft met name te maken met de complexiteit welke het statefull karakter met zich meebrengt. Het wijzigen van mailserver software of cluster software is bijna ondenkbaar zonder de architectuur binnen deze tier op nieuw te ontwerpen. Het is wel mogelijk om een derde (of eventueel meerdere), node(s) aan het cluster toe te voegen.

2.3 Tier 3

In de derde tier is de data opslag ondergebracht. Voor de opslag van de email zijn verschillende mogelijkheden. Er is de keuze omdat in verschillende formaten te doen:

mbox format: Dit formaat is het meest gebruikte formaat op traditionele Unix systemen. In dit formaat worden alle berichten opgeslagen in één grote file, dit is meestal var/spool/mail/USERNAME, of andere locaties als /var/mail/USERNAME of /usr/spool/mail/USERNAME. Berichten binnen de file worden gescheiden door de delimiter "From ".

MMDF format: Dit formaat is vergelijkbaar met het mbox formaat. Het verschil is dat de delimiter `^A^A^A^A` (4 control-A's) aan het begin en aan het einde van het bericht worden gebruikt. Dit formaat wordt traditioneel gebruikt op SCO Unix.

MH format: Het MH formaat was het eerste formaat wat de berichten als aparte files opsloeg in een directory. Elke email is een file met een numerieke naam.

Maildir format: dit formaat bewaart net als het MH formaat, de berichten als aparte files in een directory. Maildir biedt echter wat extensies op het MH formaat, wat het wat robuuster maakt. Het maildir formaat wordt veelgebruikt wanneer de storage zich op een NFS server bevindt. Omdat het geen locking nodig heeft.

Database: Een andere mogelijkheid is de berichten op te slaan in een database in plaats van in files en directories. Dit wordt vooral veel toegepast in commerciële producten, een voorbeeld hiervan is de exchange mailserver van Microsoft.

Bovenstaande opties zijn waarschijnlijk de meest voorkomende manieren om mail op te slaan. Ieder van deze opslagformaten heeft zijn eigen voor en/of nadelen. Zo kan optie X kan schaalbaarder zijn dan optie Y, terwijl optie Z weer beter een betere performance heeft. Er moet dus een wel overwogen afweging worden gemaakt.

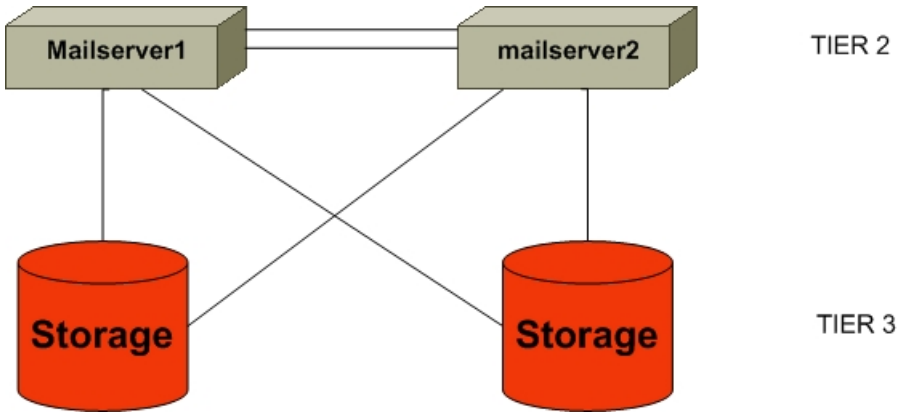
Er dient ook naar de storage zelf gekeken te worden. Er zijn hiervoor verschillende mogelijkheden, zoals NFS of direct attached storage.

2.3.1 Scalability

Het is belangrijk dat er altijd blijvend voldoende opslag capaciteit beschikbaar is. Bij het opzetten van de mail infrastructuur dient hier rekening mee gehouden te worden. Dit kan door voldoende mogelijkheden te hebben om nieuwe storage hardware aan te sluiten. Ook de keuze van het opslagformaat van de email speelt hierin een belangrijke rol.

2.3.2 Availability en Reliability

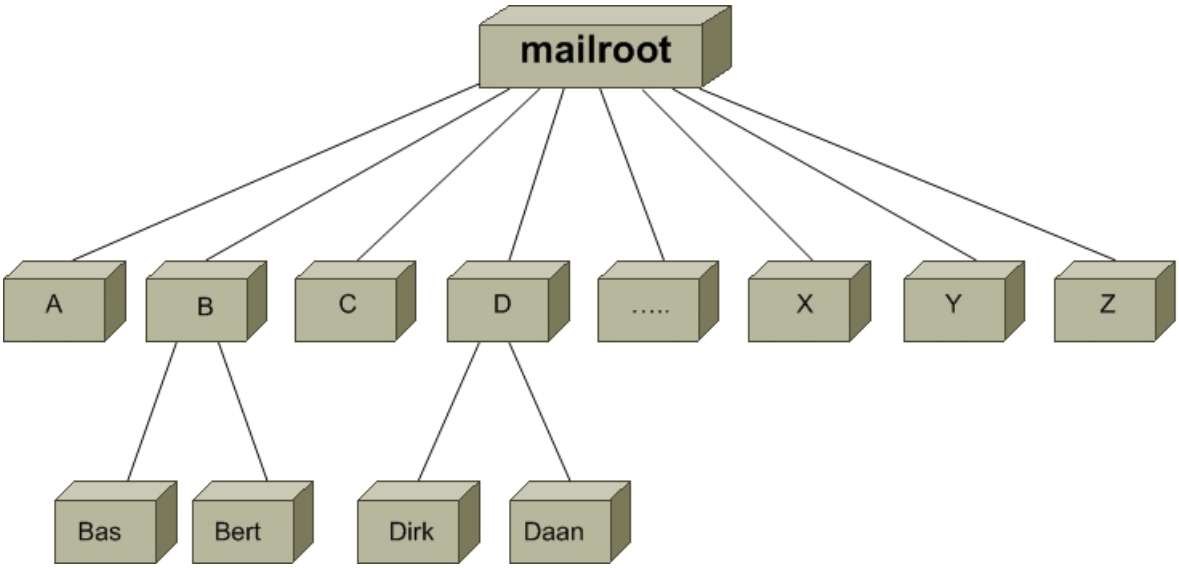
Om de availability en reliability te verhogen kon in tier 2 gekozen worden voor een cluster. Dit heeft ook gevolgen voor de storage. De storage kan onderdeel zijn van een cluster zodat deze ook een hogere availability en reliability krijgt. Een mogelijkheid om de betrouwbaarheid van het systeem te verhogen is door gebruik te maken van RAID. Dit maakt het mogelijk dat een disk kan uitvallen zonder dat er data verloren gaat. Een andere mogelijkheid is een tweede storage parallel te plaatsen aan de primaire storage array. Deze secundaire storage array is dan exacte kopie van de primaire en kan in het geval van een grote hardware storing de rol van de primaire storage array overnemen.



Figuur 2.6: *cluster met storage*

2.3.3 Performance

Schrijven naar disk is relatief langzaam, als dit verbeterd kan worden zal de performance van het hele systeem verbeterd worden. Door de storage arrays te voorzien van een grote hoeveelheid cache (bijvoorbeeld 1GB) kan de performance enorm verbeterd worden. Ook moet gelet worden op de indeling van de directories. Wanneer er in een organisatie veel gebruikers en dus mailboxen of Maildirs zijn, is het belangrijk na te denken over hoe dit in te delen. Uiteraard is het mogelijk om alle Mailboxen of Maildirs in één Directory te zetten. Het is echter aan te raden dit verdelen over meerdere directories. Een voorbeeld indeling kan zijn een indeling gebaseerd op de eerste letter van de username, zie voor een voorbeeld figuur 2.7.



Figuur 2.7: *indeling in verschillende directories*

Het is voor een operating sytem veel makkelijker zoeken naar een bestand in een directory met honderd bestanden dan in een directory met enkele duizenden bestanden. Het 'hashen"van de directories komt de performance dan ook ten goede.

2.3.4 Security

Security op dit niveau heeft vooral te maken met rechten op bestanden en files. Ook file locking speelt hier een rol. File locking wordt gebruikt wanneer een bestands formaat als mailbox wordt gebruikt. Wanneer een bestand geopend wordt, moet deze gelocked worden, Dit om te voorkomen dat er 2 schrijfacties o dezelfde file gebeuren, met als gevolg dat de file (en dus mailbox) corrupt raakt. File locking speelt geen rol bij het gebruik van maildir formaat.

2.3.5 Manageability

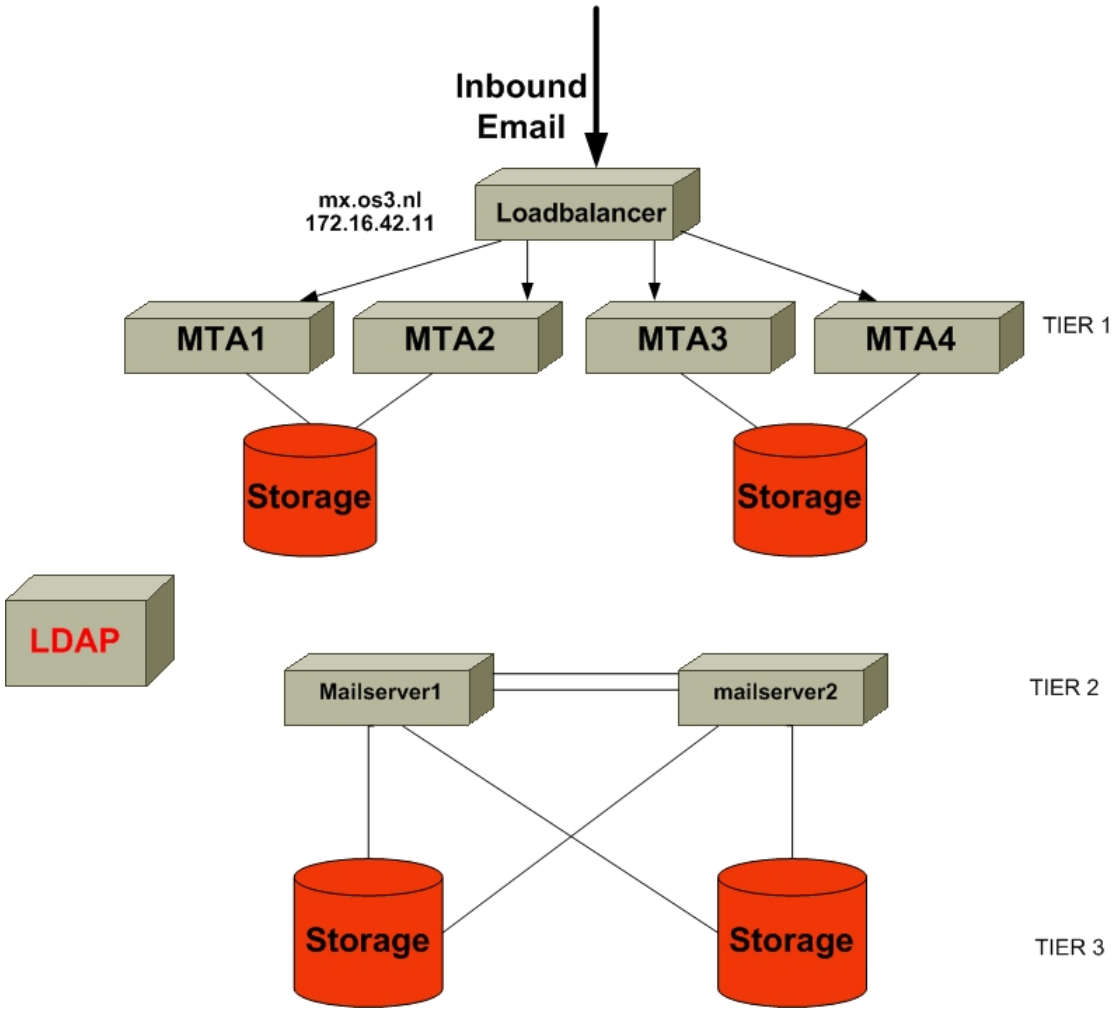
Omdat de storage de meest belangrijkste keten uit het systeem is (dit is namelijk waar alle mailcontent staat) is deze vaak redundant uitgevoerd. Het toevoegen van complexiteit als Raid/clustering maakt het over het algemeen wat moeilijker te beheren. Echter de meeste producten en Operating systems komen met een aantal handige beheers tools en maken het beheer weer eenvoudiger. Voor het operating system lijkt een raid cluster op een normale "logische"schijf, deze heeft dus in principe geen kennis van de complexiteit. Typische beheers zaken zijn: het aanmaken van nieuwe gebruikers en dus mailboxen/dirs. Het kan ook voorkomen dat mailboxen corrupt raken, op dat moment zal een Mailbox hersteld moeten worden, over het algemeen zijn hier bijgeleverde tools voor.

2.3.6 Adaptability

In productie situaties zullen over het algemeen weinig wijzigingen gedaan worden. Een wijziging die uit performance en schaalbaarheids overwegingen gemaakt kan worden is een migratie naar een ander opslag formaat. Een overgang van Mailbox naar Maildir formaat is mogelijk maar moet zeker wel overwogen gedaan worden. Het wijzigen een opslagformaat heeft gevolgen voor alle services welke hier gebruik van maken. De mailboxen zullen geconverteerd moeten worden naar Mailbox formaat en de applicatie laag zal het nieuwe formaat moeten ondersteunen.

2.4 Voorbeeld architectuur

Een aanbevolen voorbeeld architectuur zou eruit kunnen zien als in figuur 2.8. Dit is een voorbeeld architectuur voor de inbound email. Inkomende email komt aan bij de mx host voor het domein mx.os3.nl. Dit IP-adres is geconfigureerd op de loadbalancer (welke ook weer redundant opgezet kan zijn). De loadbalancer verdeeld het SMTP verkeer over één van de vier frontend MTA's. Deze plaatsen de mail in de mailqueue zodat een ander proces de mail verder kan afhandelen. Voor ieder emailbericht dienen één of meerdere files naar disk te worden geschreven. Bijvoorbeeld: statusfile, headerfile en de emailcontent. Het schrijven van iedere tijdelijk file naar disk, kost een aantal synchrone I/O calls. Om deze performance te verbeteren wordt gebruik gemaakt van storage arrays welke over en grote hoeveelheid cache beschikken. De meeste data zal hoogstwaarschijnlijk nooit fysiek naar disk worden geschreven omdat deze meestal maar een aantal honderd milliseconde bestaan. (Van de grote hoeveelheid beschikbare storage wordt verder geen gebruik gemaakt). Het volgende proces zal de mail weer uit de mailqueue halen en verder verwerken. Eventueel kan het bericht nu op virussen en/of spam gechecked worden. Vervolgens wordt er gekeken wat er nu verder met de email moet gebeuren. Hiervoor worden diverse LDAP queries gedaan bij de LDAP server. Er zal worden opgevraagd op welke van de twee mailservers de mail voor deze gebruiker afgeleverd dient te worden. De mail wordt nu doorgestuurd naar mailserver1 of mailserver2. Deze SMTP



Figuur 2.8: *SMTP voorbeeld architectuur*

server heeft toegang tot de mailbox van de gebruiker. Wanneer het email bericht hier binnen komt zal deze weer in de mailqueue geplaatst worden. Hier speelt de cache van de mailstorage weer een belangrijke rol om de performance te verbeteren. Ook deze mailserver zal weer een LDAP query doen om te achterhalen waar de mailbox van de gebruiker zich bevindt. Er staan diverse parameters over een gebruiker met betrekking tot email in de LDAPserver gedefinieerd, een voorbeeld is:

```
dn: uid=andree,ou=people,o=os3,c=nl
mailHost: mailserver1.srv.os3.nl
sn: Toonk
cn: Andree Toonk
uid: andree
mail: andree@os3.nl
mailMessageStore: /mailroot/a/
givenName: Andree
mailQuota: 10485760
```

In dit voobeeld, is de user andree (dn: uid=andree,ou=people,o=os3,c=nl) alleen bekend op mailserver1en de mailbox voor deze gebruiker bevind zich in de directory /mailroot/a/. Ook het mail quota van deze gebruiker is gedefinieerd in de LDAP server. De mailservers op tier 2 alsmede de storage op tier 3 zijn in een cluster opgenomen. Dit resulteert in een hogere availability en reliability omdat componenten in een cluster in staat zijn elkaars functie over te nemen. De eisen welke aan het filesystem gesteld worden zijn op de backend hoger dan die aan de frontends gesteld worden.

Hoofdstuk 3

POP en IMAP

POP3 en IMAP4 zijn internetmailservices welke gebruikers in staat stellen hun mail op te halen en te raadplegen. In grote ICT-omgevingen werd in het verleden vooral POP aangeboden vanwege de eenvoud en de bescheiden eis welke er aan de POP servers gesteld wordt. Zo kon met veel gebruikers op een centraal mailsysteem toch een redelijke performance behaald worden. Met POP kan mail alleen opgehaald worden, mailberichten worden tijdens dit ophalen veelal meteen van de POPserver verwijderd. Meerdere mailfolders op de POPserver aanmaken is niet mogelijk.

IMAP wordt steeds populairder doordat het meer functionaliteiten en features biedt dan POP. Zo kan de gebruiker meerdere mailfolders aanmaken en mail gestructureerd in verschillende folders opslaan. De mail blijft hierbij bewaard op de server. Dit heeft als voordeel dat mail nu op een centrale plaats opgeslagen blijft en vanaf meerdere locaties toegankelijk is. Ook door de toenemende populariteit van webmail waarbij de webbrowser als mailclient fungeert, is IMAP als basis daarvan een services die steeds meer gebruikt wordt. IMAP genereert echter een veel hogere load op de servers dan POP. Dit vereist een zwaardere aanpak en resulteert in zwaardere hardware voor de IMAP servers. Verder is de eis welke aan de dataopslag wordt gesteld significant hoger omdat mail op de server opgeslagen blijft.

Gekeken wordt naar het opzetten van de POP en IMAP infrastructuur voor een groot aantal gebruikers. Hierbij zullen POP en IMAP gezamenlijk behandeld worden. In de praktijk is het gebruikelijk om een enkele mailbox met zowel POP als IMAP uit te lezen. De mail storage is dan voor beide services toegankelijk. POP en IMAP zullen verder als één gezien worden vanwege de vele overeenkomsten welke ze binnen een infrastructuur hebben waarbij IMAP bepalend is voor de belasting van het systeem.

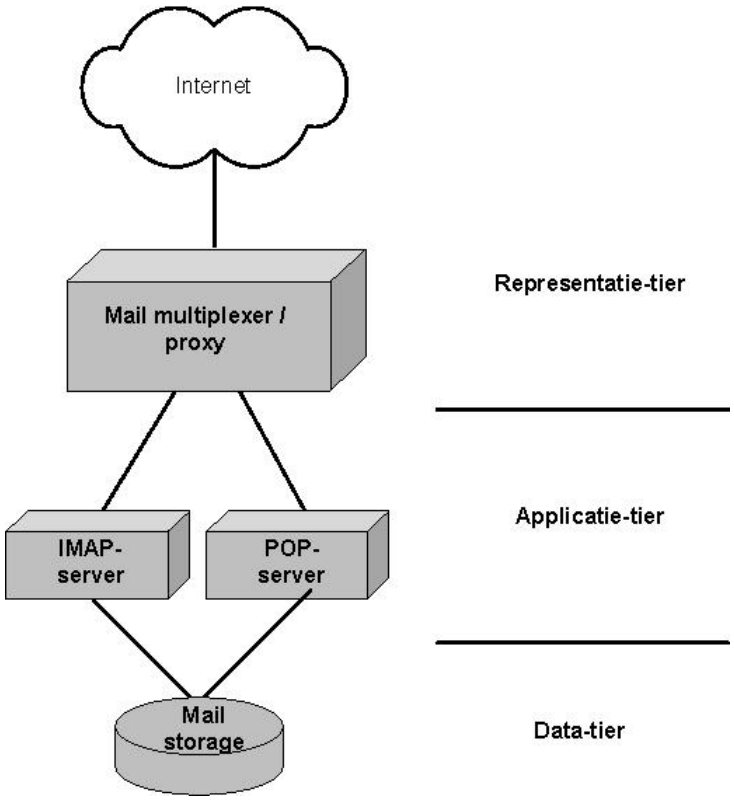
Net als bij SMTP en WWW zijn POP en IMAP uiteen te zetten volgens de 3-tier architectuur:

- Front-end proxy (representatie-tier)
- IMAP/POP server (applicatie-tier)
- Mail storage (data-tier)

Figuur 3.1 laat de 3-tier architectuur van POP en IMAP zien met de onderlinge samenhang. Iedere

tier zal afzonderlijk behandeld worden. Per tier zullen de in hoofdstuk 1 genoemde punten nader bekeken worden, te weten:

- Scalability
- Performance
- Availability en reliability
- Security
- Manageability
- Adaptabililty

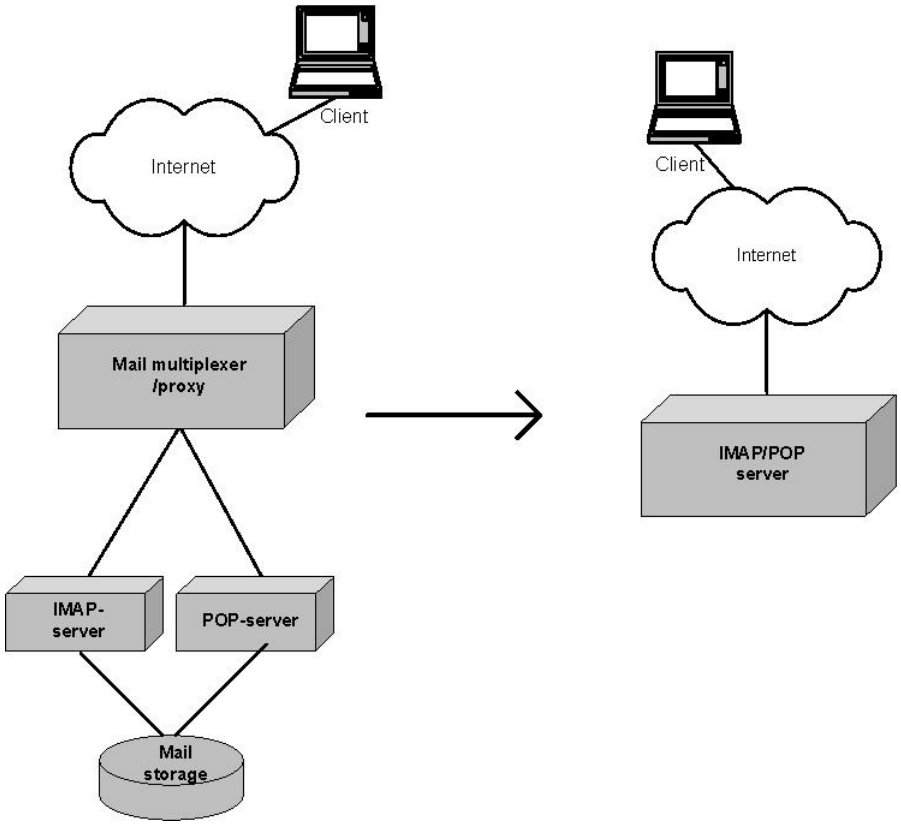


Figuur 3.1: *3-tier POP en IMAP*

3.1 Front-end proxy (representatie-tier)

Met de front-end wordt de koppeling of interface naar de gebruikers gerealiseerd. Deze laag fungeert als proxy/mail-multiplexer voor de connecties tussen de gebruiker en de juiste IMAP/POP server. Deze laag biedt een simpele interface waardoor client-configuraties voor de gebruikers eenvoudig blijven. De front-end zorgt ervoor dat geen er kennis noodzakelijk is van de mail-infrastructuur (zie figuur 3.2). Vooral bij grote netwerken met veel gebruikers kan de mail-infrastructuur door

bijvoorbeeld multi-host mailsystemen erg ondoorzichtig en complex zijn. De frontend schermt deze complexiteit voor de gebruiker af. De eenvoud van de aan de gebruikers aangeboden services dient zoveel mogelijk gestimuleerd te worden. Uniforme login informatie en identieke configuraties voor alle gebruikers kunnen hier aan bijdragen. Hierdoor is kostenbesparing mogelijk op de bezetting van een helpdesk.

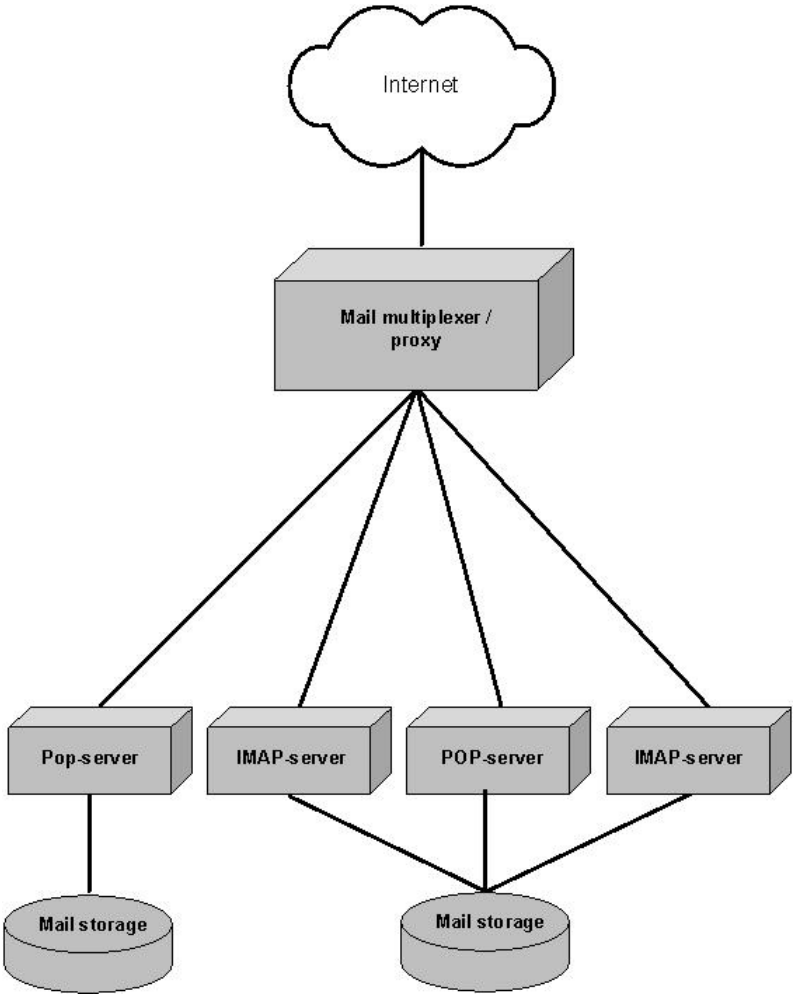


Figuur 3.2: *Verborgen complexiteit voor gebruikers*

3.1.1 Scalability

De functie van de proxy is het bepalen van de juiste server en daarna het verzorgen van een bi-directionele connectie tussen de gebruiker en de betreffende IMAP/POP server. Het bepalen van de juiste IMAP/POP server kan op twee manieren gebeuren. Het kan met rules waarbij door de proxy wordt gekeken naar bijvoorbeeld een deel van de inlognaam en afhankelijk van de eerste letter bepaald wordt naar welke IMAP/POP server de gebruiker doorgestuurd zal worden. Ook kan de inlognaam een emailadres zijn, waarbij de proxy afhankelijk van het domein van het emailadres de gebruiker doorstuurt naar de juiste IMAP/POP server. De tweede manier is met een directoryserver zoals LDAP te gaan werken waarvan in de database voor de individuele gebruiker staat wat de juiste IMAP/POP server is. De proxy kijkt voor de betreffende gebruiker in de database en bepaalt zo de juiste IMAP/POP server. Het voordeel hiervan is de flexibiliteit waarmee een gebruikersmailbox op een willekeurige IMAP/POP server kan worden geplaatst. De proxy zorgt verder voor de schaalbaarheid van de applicatie-tier. Een toevoegen van een geheel afwijkend mail systeem aan de applicatie-tier kan eenvoudig door de front-end opgevangen worden en heeft zo een integrerende

funktie van verschillende mailsystemen. Multiplexing/redirection naar verschillende back-end mail-servers, partitioning/splitting van usermailboxes zijn eigenschappen welke de schaalbaarheid van de applicatie tier vergroten (zie figuur 3.3). De proxy zelf is door het stateless karakter makkelijk horizontaal schaalbaar. Door LDAP te gebruiken voor het lokaliseren van de juiste mailserver voor de betreffende gebruiker, is het parallel bijplaatsen van identieke mail-multiplexer eenvoudig te realiseren.



Figuur 3.3: *Schaalbaarheid door de proxy*

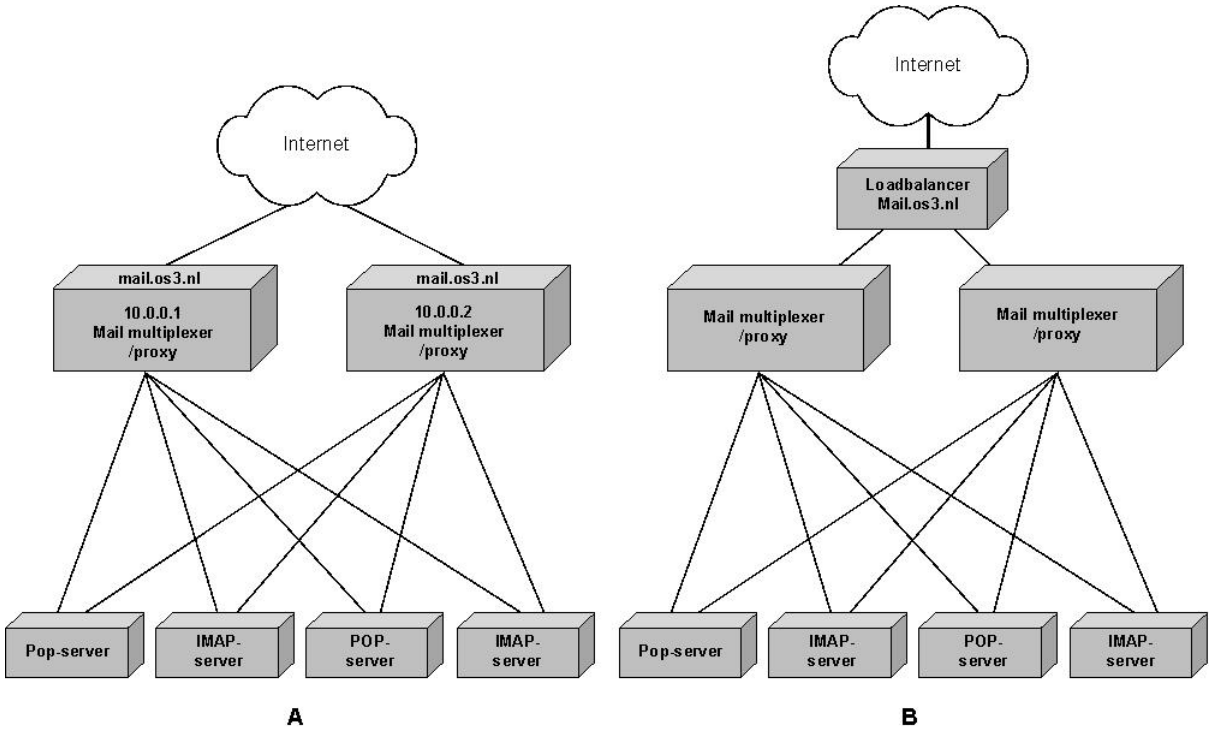
3.1.2 Performance

De performance van de front-end is te verhogen door meerdere proxies parallel te plaatsen. Op deze proxies kan loadbalancing toegepast worden waarbij achter een enkele virtuele proxy meerdere fysieke proxies worden geplaatst. De load van mail-multiplexers is vanwege het "pass-through"karakter niet hoog. Deze belasting is afhankelijk van het aantal parallelle connecties van gebruikers dat op de verschillende back-end mailservers geserveerd moet worden. Bij gebruik van een directory server zoals LDAP, is de bottleneck in de performance meestal de lookups naar deze server. Deze lookup moet bij elke IMAP/POP connectie plaatsvinden. Performance hiervan kan verhoogd worden door

dagelijks een hash van de database met mappings tussen gebruikers en mailservers op de lokale proxies te plaatsen. Hierdoor vindt er niet voor iedere clientconnectie een request naar de directory server meer plaats, maar wordt er lokaal in de cache gekeken. Het nadeel is dat wijzigingen van gebruikersgegevens niet real-time doorgevoerd worden omdat de directory server zelf niet geraadpleegd wordt.

3.1.3 Availability en reliability

De proxy is een single point of failure en dient daarom redundant uitgevoerd te worden. Door loadbalancing toe te passen wordt deze redundantie ingebouwd . Dit is mogelijk in de vorm van round robin DNS, waarbij iedere mail-multiplexer een eigen IP-adres heeft (zie figuur 3.4a). Een DNS-requests van de client naar mail.os3.nl zal resulteren in het afwisselend IP-adres van de verschillende proxies. Een andere oplossing is loadbalancing op basis van layer4 switches. Hierbij vindt multiplexing plaats op inkomende TCP/IP connections. Loadbalancing kan eveneens gerealiseerd worden met vendor specifieke dedicated hardware oplossingen (zie figuur 3.4b). Ook bestaan er softwarematige oplossingen zoals Linux Virtual Server. Net als een enkele proxy, is een enkele loadbalancer ook een single point of failure en moet redundant uitgevoerd te worden.

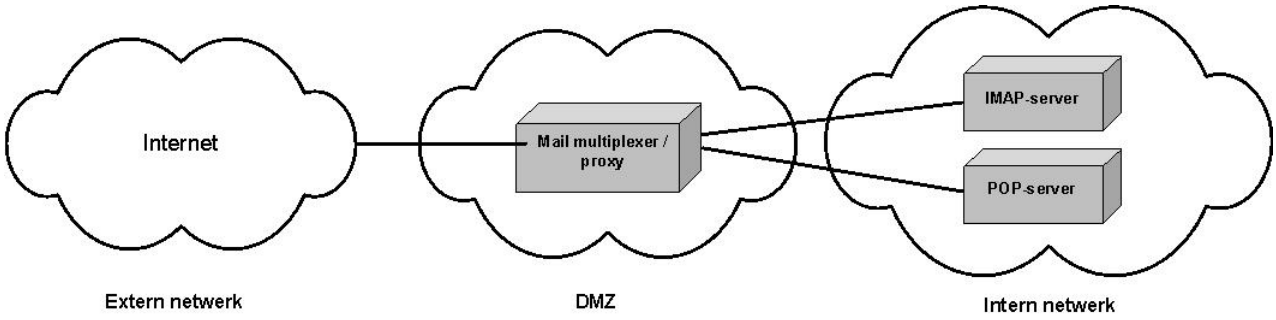


Figuur 3.4: *Redundantie en loadbalancing van de proxy*

3.1.4 Security

De proxy kan in een DMZ (Demilitarized Zone) geplaatst worden. Deze zone is een tussenliggend netwerk tussen het publieke onveilige netwerk met de clients en het interne veilige servernetwerk met de IMAP/POP servers (zie figuur 3.5). Deze IMAP/POP servers staan alleen toegang vanaf

de DMZ toe en niet vanaf het externe netwerk. De proxy zelf kan fungeren als firewall machine. Alle client-connecties voor POP en IMAP komen hierop binnen, dus mail-policies kunnen centraal opgelegd worden. Omdat de proxy redundant en schaalbaar opgesteld is, kan firewalling op de proxies extra beheerslast geven. Firewalling of filtering kan ook elders op één plaats voor de hele DMZ geïmplementeerd worden. Netwerk-policies kunnen zo centraal voor de gehele infrastructuur toegepast worden.



Figuur 3.5: *DMZ*

Bij zowel POP als IMAP worden logingegevens van de gebruiker als plain-text van de client naar de server verzonden. Dit is een beveiligingsprobleem. Inlognamen en wachtwoorden kunnen onderschept worden. Een oplossing hiervoor is de verbinding tussen de client en de server encrypted op te zetten door middel van SSL[1]. De proxy handelt de server-kant af voor SSL. De IMAP/POP servers zelf hoeven hierbij geen SSL-ondersteuning te bieden, omdat deze niet rechtstreeks met de clients maar met de proxy een verbinding hebben. Deze verbinding gaan over een intern netwerk. Het is voor kwaadwillenden hier niet mogelijk om logingegevens te onderscheppen. Voorwaarde voor deze encryptie is dat de gebruikers een mailclient gebruiken welke SSL ondersteund. Hoewel IMAP-SSL en POP-SSL steeds meer worden toegepast, worden POP en IMAP zonder SSL nog het meest gebruikt. De proxy fungeert voor de IMAP/POP servers als buffer voor aanvallen vanaf het publieke netwerk en zorgt dus voor een hogere security. Voor aanvallen zoals DDoS[2] zal de mail infrastructuur minder gevoelig zijn.

3.1.5 Manageability

Beheer van de proxies/multiplexers is niet intensief. Het beheer zal voornamelijk bestaan uit het monitoren van aanvallen op de front-end zoals DoS. Als er met een directory server gewerkt wordt, bevatten de proxy/multiplexers geen lokale data. Met loadbalancing is het inzetten van extra proxies beheertechnisch gezien eenvoudig te realiseren. Ook onderhoud aan operationele proxies is makkelijk te doen door de betreffende proxy eerst uit de loadbalancer-pool te halen.

3.1.6 Adaptabililty

De internetmailservices POP3 (RFC 1939) en IMAP4(RFC 3501) zijn gebaseerd op open internetstandaarden. Dit is belangrijk voor het modulair kunnen opbouwen dat door middel van het verticaal opsplitsen in meerdere tiers wordt bereikt. De proxy/multiplexing functie is niet een standaard

[1] Secure Socket Layer
[2] Distributed Denial of Service attack

service met een RFC. Het is een extra service welke tussen de client en de server staat. Doordat de IMAP/POP servers zijn gebaseerd op open standaarden, is het mogelijk om een proxy/multiplexer te ontwikkelen. Deze kan geoptimaliseerd worden voor de eigen specifieke ICT-infrastructuur. Een voorbeeld van optimalisatie is het monitoren van de load op IMAP/POP servers en afhankelijk daarvan kan de proxy nieuwe IMAP/POP connecties aan een server met een lage load koppelen. Vulnerabilities welke van toepassing zijn op veelgebruikte vendor proxies, zullen minder van invloed zijn op een eigen ontwikkelde proxy. Veelal worden vendor oplossingen gebruikt. Een belangrijke reden hiervoor is dat op het product support geleverd wordt. Er dient gerealiseerd te worden dat als de proxy faalt door een softwarefout, de gehele IMAP/POP service wegvalt.

3.2 IMAP/POP server (applicatie-tier)

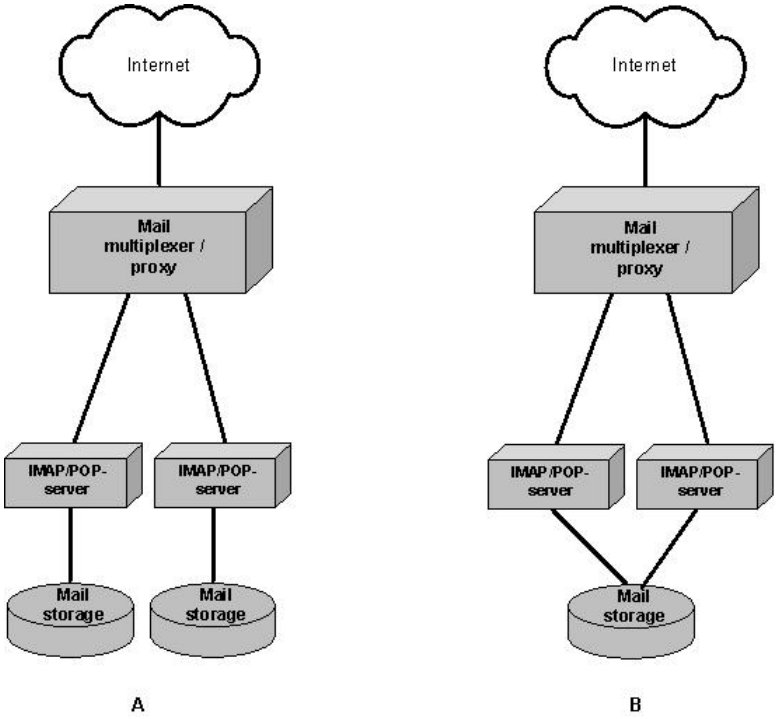
De applicatie-tier bevat de IMAP/POP servers. Deze servers verlenen de daadwerkelijke toegang tot de mailboxen van de gebruikers. De gebruiker komt via de proxy op de juiste IMAP/POP server terecht. Omdat de proxy geheel transparant is, is er geen verschil voor de IMAP/POP server tussen een gebruiker welke rechtstreeks de server benaderd of dat de gebruiker via de proxy komt. De IMAP/POP server kan voor authenticatie gebruik maken van een lokale database of van een directory server zoals LDAP.

3.2.1 Scalability

Een veel gebruikte manier van schalen van mail is het partitioneren van mailboxen. Dit houdt in dat een IMAP/POP server over een decicated mail storage beschikt (DAS, SAN) welke niet met andere servers dan de MTA wordt gedeeld (zie figuur 3.6a). De MTA ontvangt de mail en bezorgt deze door het op de mail storage op te slaan. De IMAP/POP service bevindt zich op dezelfde fysieke server als de MTA. Beide hebben toegang tot dezelfde mail storage. De IMAP/POP server haalt de mail van de mail storage op voor de gewenste gebruiker. Het maximaal aantal mailboxen dat een IMAP/POP server acceptabel kan verwerken is beperkt. De belasting van de IMAP/POP servers loopt op bij een groot aantal mailboxes dat tegelijkertijd door gebruikers wordt benaderd. Om deze belasting te verlagen kan een tweede IMAP/POP server met een eigen mail storage en MTA parallel gezet worden. De mailboxen kunnen zo over meerdere servers worden verspreid. Deze aanpak maakt mail schaalbaar. Een nadeel is dat mailboxes over verschillende mail storages verdeeld wordt. Zo zijn gedeelde folders in IMAP niet meer mogelijk. Het kan nodig zijn dat mailboxen verplaatst moeten worden om de load van de servers beter te verdelen of om het diskgebruik efficiënter te benutten. Een niet veel gebruikte oplossing voor dit nadeel is om een gedeelde mail storage te gebruiken voor alle IMAP/POP servers door middel van NAS zoals NFS (zie figuur 3.6b). De IMAP/POP servers zijjn daardoor horizontaal schaalbaar geworden. Een voorwaarde voor de gedeelde mail storage is dat er gebruik gemaakt dient te worden van een mailopslag formaat welke geen file locking problemen met zich meebrengt.

3.2.2 Performance

Vooral IMAP zorgt voor een zware belasting van de servers. Zoals gezien, geeft het schalen door meerdere statefull IMAP/POP servers naast elkaar te plaatsen om de performance te verhogen de nodige problemen. Beter kan er eerst gekeken worden naar zwaardere server-hardware. Net als bij de proxies, kan het gebruik van directory servers performanceverlies tot gevolg hebben. IMAP/POP



Figuur 3.6: *Dedicated of gedeelde mail storage*

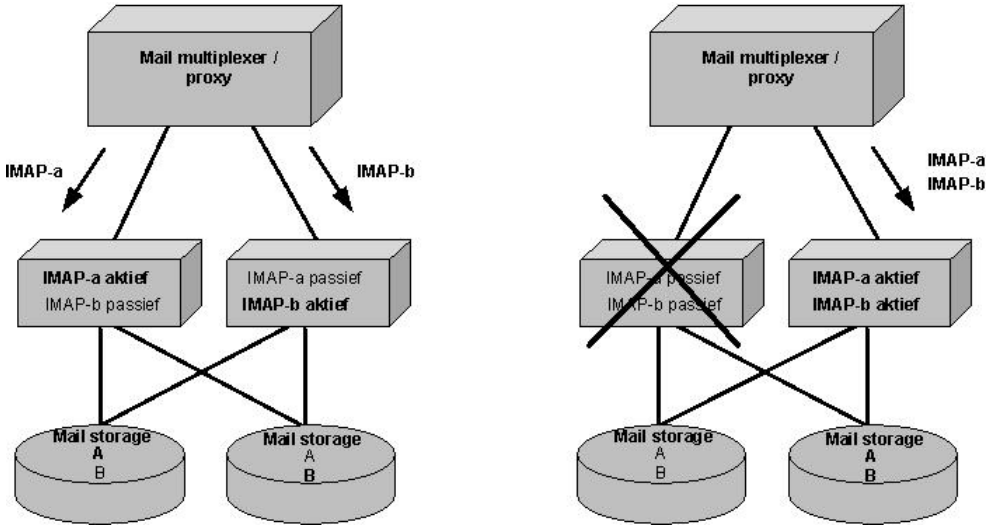
en MTA servers zijn systemen welke hoge I/O eisen stellen aan files access. Door storage arrays te gebruiken met veel cachegeheugen (1Gbyte) wordt aan die eisen tegemoet gekomen. Het gebruik van NFS geeft bij mailsystemen een erg slechte performance. Er wordt over het LAN gebruik gemaakt van een filesharingprotocol dat vooral slecht overweg kan met kleine bestanden. Bovendien maken er meerdere IMAP/POP servers gebruik van een NFS-fileserver. Dit geeft extra performanceverlies en ook file-locking problemen. Deze nadelen zullen afgewogen moeten worden tegen de eerder genoemde voordelen van NFS.

3.2.3 Availability en reliability

Aan de uptime van IMAP en POP services worden hoge eisen gesteld. Het down gaan van een server mag geen gevolgen hebben voor de gebruikers. Daarom dienen statefull IMAP/POP services redundant opgezet te zijn. Dit wordt gerealiseerd door clustering waarbij de services dubbel op verschillende hardware worden opgezet. Een van de twee service is actief, de andere service staat standby en wordt actief na uitval. Voor de disks kan mirroring worden toegepast. Om kosten te drukken, kan kruislings op beide servers een willekeurige service actief zijn waarbij de andere server in het cluster voor die service standby staat (zie figuur 3.7). Het nadeel is dat er bij failover twee (of meer) services op een enkele server actief zijn. Dit komt de performance niet ten goede.

3.2.4 Security

De IMAP/POP servers staan in een intern veilig netwerk en zijn alleen benaderbaar door de proxies. Hierdoor zijn de servers relatief veilig voor aanvallen van buitenaf. Overwogen kan worden om op



Figuur 3.7: *IMAP/POP server geclusterd*

de proxy naast het bepalen van de juiste IMAP/POP server ook de authenticatie van gebruikers uit te laten voeren. Bij gebruik van een directory server kan door de proxy volstaan worden met een enkele lookup. De IMAP/POP server vertrouwt vervolgens de geauthenticeerde connecties welke afkomstig is van proxy en voert zelf niet opnieuw authenticatie uit voor de gebruiker. De IMAP/POP server vertrouwt dus iedere gebruiker. Het voordeel is dat de IMAP/POP servers ontlast worden van authenticatie. Toch moet overwogen worden of dit voordeel opweegt tegen de hogere security risico's. Als een indringer de proxy weet te omzeilen, dan geeft de IMAP/POP server deze indringer toegang tot alle mailboxen.

3.2.5 Manageability

Het beheer van redundant opgezette servers is tamelijk complex. De aanpak van clustering vergt inzicht van de beheerders. Het is belangrijk dat failover regelmatig getest wordt. Als de IMAP/POP servers stateless zijn door een gedeelde mail storage te gaan gebruiken is clustering niet nodig en zijn de IMAP/POP servers net als de proxies horizontaal schaalbaar. Verder moeten de servers gemonitord worden. Er kunnen alerts verzonden worden als bepaalde niveaus worden overschreden of als een bepaalde status wijzigt.

3.2.6 Adaptabililty

Door met standaard opslagformaten zoals mbox en maildir te werken voor de mail storage, is het in principe eenvoudig om over te gaan op een andere IMAP/POP server. Er zal getest moeten worden of veel gebruikte IMAP/POP clients geen problemen ondervinden met de nieuwe IMAP/POP server doordat deze clients kunnen afwijken van de RFC's.

3.3 Mail storage (data-tier)

De mail storage bevindt zich op de data-tier en bevat de maildata van de gebruikers. Mailsystemen veroorzaken veel I/O operaties op de disks. Niet alleen de mailspool maar ook de mail storage wordt met veel lees en schrijf handelingen belast. Een MTA ontvangt mail en slaat dit voor lokale bezorging op in een mail storage. De gebruiker heeft met behulp van een mailclient toegang tot deze mail via een IMAP/POP server. De IMAP/POP server leest vooral de mail storage, terwijl de MTA alleen maar schrijft. Met POP wordt mail alleen maar gelezen en verwijderd, in feite de tegenovergestelde functie van de MTA. Hoewel het met POP mogelijk is om de mail op te halen zonder deze te verwijderen, kunnen gebruikers door een beperkte opslagruimte gedwongen worden de mail met het ophalen meteen van de mail storage te verwijderen. De MTA zal geen mail bezorgen in een te volle mailbox en kan ontvangende mail bijvoorbeeld terugsturen naar de betreffende zender met een foutmelding dat afleveren niet mogelijk is. Hiermee kan de beperkte en dure opslagcapaciteit binnen bepaalde grenzen gehouden worden. Een maximum mailboxruimte per gebruiker van 20 Mbyte is voor 10.000 gebruikers al bijna 200 Gbyte aan dataopslag vereist. Met het gebruik van IMAP waarbij de mail niet verwijderd wordt, neemt de opslagcapaciteit sterk toe. Dit kan een argument zijn om alleen POP aan te bieden. Het aantal leesacties op de mail storage door de IMAP-server is veel hoger dan bij POP. Doordat gebruikers een lokale kopie van de mail hebben, wordt de mail storage enigszins ontlast, mail wordt dan eenmalig opgehaald en lokaal op de client als kopie bewaard. Ook schrijfacties vinden er meer plaats door bijvoorbeeld het verplaatsen van mail tussen verschillende folders. Mail kan in verschillende formaten opgeslagen worden. Globaal gezien zijn er drie verschillende types:

- Mbox
- Maildir
- Database

Met Mbox wordt de mail per gebruiker in een enkele file opgeslagen. Deze file kan erg groot worden. Als deze file corrupt raakt doordat bijvoorbeeld twee processen tegelijk schrijven als gevolg van onbetrouwbare file-locks, is de gehele mailbox onbruikbaar geworden.

Maildir slaat ieder mailbericht in een aparte unieke file op waardoor file-locking overbodig wordt. Het is dus niet mogelijk dat twee processen naar dezelfde file schrijven, zoals dit gebeurd bij het Mbox formaat. Bij het Mbox formaat kan het gebeuren dat er door de client gelezen wordt terwijl de MTA op hetzelfde moment in dezelfde file probeert te schrijven. Bij Maildirs doet dit probleem zich niet voor. Daardoor kan NFS gebruikt worden. Een nadeel van Maildir is het inefficiënte gebruik van diskruimte. Als de storage gebruik maakt van 64 Kbyte blokken voor de opslag van files waarbij een blok maximaal één file bevat en de gemiddelde grootte van de mailberichten 4 Kbyte is, geeft dit een verspilling van 90% diskruimte. Ook de interne file-tabellen worden door het aantal files erg groot.

Een database leidt tot een complexere benadering van de MTA en de IMAP/POP server waarbij niet rechtstreeks naar een file wordt geschreven of gelezen. Er is kennis vereist van hoe de database aangesproken moet worden. Het voordeel is dat file-locking hier niet speelt, de database zorgt voor de correcte afhandeling van parallelle processen.

3.3.1 Scalability

Het uitbreiden van mail storage capaciteit kan op verschillende manieren aangepakt worden. Dit hangt af van de aanpak welke met de applicatie-tier is behandeld:

- Eén centrale mail storage voor alle IMAP/POP servers
- Eén mail storage per IMAP/POP server

Het uitbreiden van een bestaande mail storage is niet dynamisch te realiseren. De uitbreiding van de opslag capaciteit kan gerealiseerd worden door het overzetten van de data op een grotere partitie. Een andere manier is het bijzetten van een extra partitie welke als directory wordt gekoppeld aan het file system van de bestaande mail storage. Dit laatste is niet in elke situatie mogelijk, een database leent zich bijvoorbeeld hier niet voor. Er moet zowel voor de MTA als voor de IMAP/POP server van iedere mailbox een locatie worden bijgehouden. Hiervoor kan gebruik gemaakt worden van een directory server voor het opvragen van de locatie. Als er gebruik wordt gemaakt van een mail storage per IMAP/POP server, is schaalbaarheid eenvoudig te realiseren door het bijplaatsen van meerdere mail storages en de daaraan gekoppelde IMAP/POP servers. Hiervoor is het van belang dat de proxies weten welke mailbox op een aan de IMAP/POP server gekoppelde mail storage staat.

3.3.2 Performance

Een enkele centrale mail storage welke over het netwerk wordt gedeeld, zal slechter performen dan meerdere mail storages welke een ieder direct aan een MTA en een IMAP/POP server zijn gekoppeld. De disktoegang zorgt in de meeste gevallen voor de bottleneck, en niet de applicaties zelf. Door met meerdere en kleinere mail storages te werken, kan de performance aanzienlijk worden verhoogd. Ook disk-caching werkt hier aan mee.

3.3.3 Availability en reliability

De mail storage dient net als alle andere data redundant te worden uitgevoerd en kan met RAID worden gerealiseerd. Verder kan door mirroring de mail storage over verschillende hardware verspreid worden, zodat maildata beschikbaar blijft als een storage array in het geheel uitvalt.

3.3.4 Security

De mail storage is alleen toegankelijk door de MTA en IMAP/POP servers van de applicatie-tier. Beveiliging van deze servers zorgt voor de veiligheid van de mail storage. Als er met één centrale mailstorage gewerkt wordt welke door meerdere servers gedeeld wordt, houdt dit in dat er een hoger risico bestaat dat data voor kwaadwillenden via een van de servers toegankelijk wordt.

3.3.5 Manageability

Het beheer van de mail storage kan bestaan uit het migreren van mailboxen naar de verschillende mail storages om IMAP/POP server te ontlasten en diskopslag efficiënter te benutten. Er dient geanticipeerd te worden op het groeiend aantal mailboxen zodat tijdig de capaciteitsbeperkingen van dataopslag worden opgelost.

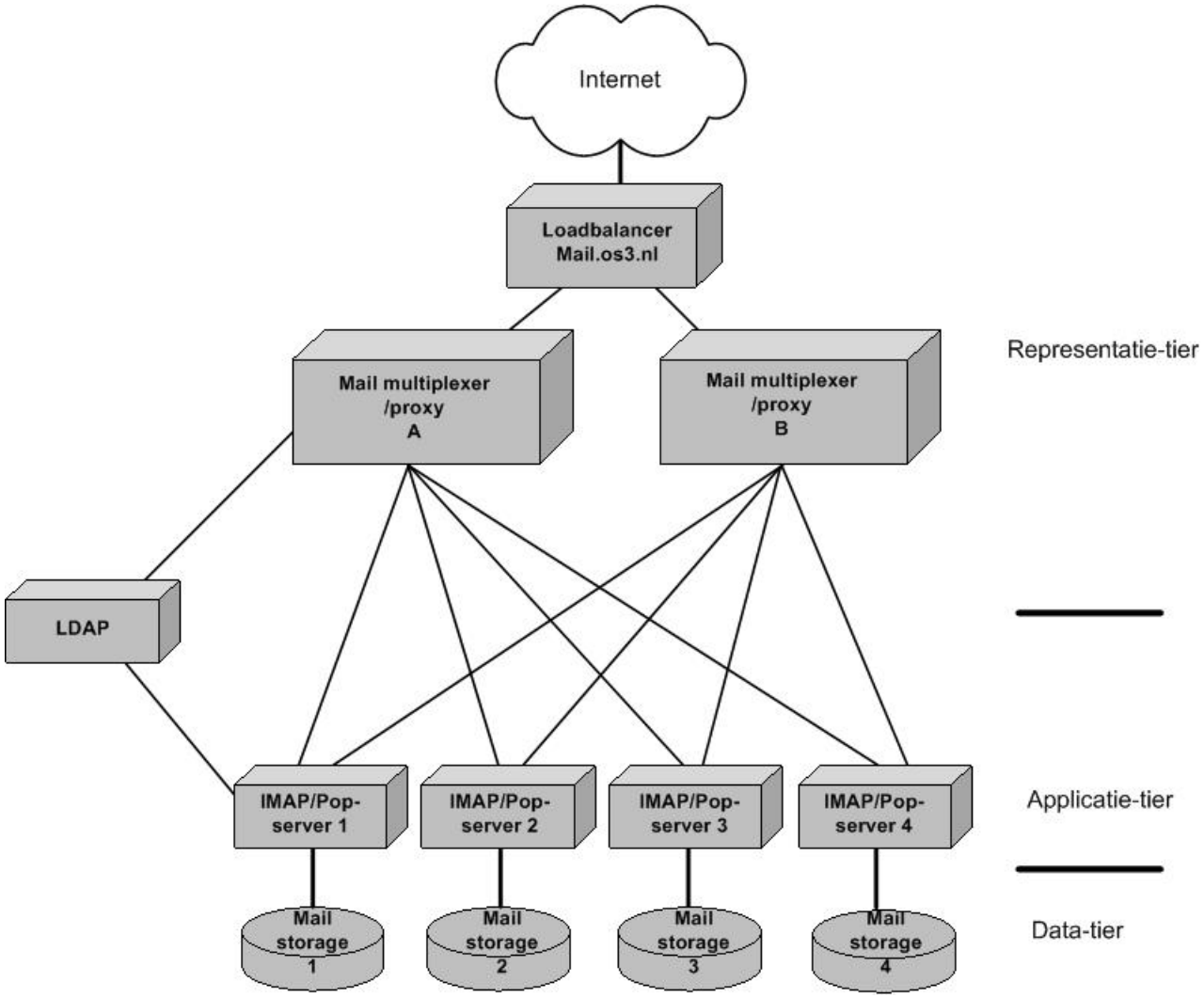
3.3.6 Adaptabililty

Als er gebruik wordt gemaakt van een database, moeten er MTA, POP en IMAP servers zijn welke met deze database kunnen communiceren. Hiervoor is geen standaard beschikbaar. Daarom zijn veel database oplossingen commercieel van aard. Bij het gebruik van een commercieel product is het verstandig om te kijken naar de mogelijkheid van exporteren van de maildata naar een bekend formaat om niet afhankelijk te zijn van een vendor zodat er in de toekomst zonder veel moeite gemigreerd kan worden naar een ander product.

3.4 Voorbeeld architectuur IMAP/POP

Een voorbeeld van een IMAP/POP architectuur zou er in de praktijk volgens figuur 3.8 uit kunnen zien.

Mail wordt door de gebruikers met POP of IMAP opgehaald vanaf de host mail.os3.nl. Deze host is een loadbalancer welke de inkomende connecties van de gebruikers verdeelt over twee proxies. De gebruiker wordt door de loadbalancer doorgestuurd naar een van de twee proxies. De betreffende proxy vraagt aan de LDAP-server welke IMAP/POP server de mailbox van de betreffende gebruiker aanbiedt. Als de mailbox van de gebruiker zich op de mail storage 3 bevindt, zal de proxy de connectie van de gebruiker door koppelen naar IMAP/POP server 3. Deze server geeft de gebruiker na authenticatie met de LDAP-server toegang tot de juiste mailbox. De proxies zijn redundant uitgevoerd en worden beide gebruikt zodat de load wordt verdeeld. Een mail storage is direct aan een IMAP/POP server gekoppeld. De mailboxes zijn over meerdere mail storages verdeeld om zo de belasting van de IMAP/POP servers te speiden. Om het voorbeeld overzichtelijk te houden is de redundantie van de IMAP/POP servers en de mail storages achterwege gelaten. Deze dienen alle vier redundant uitgevoerd te worden zoals figuur 3.7 laat zien.

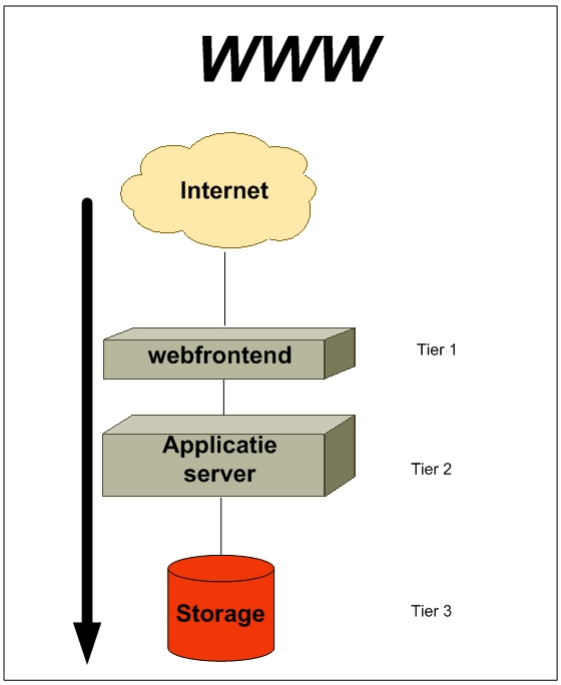


Figuur 3.8: *Voorbeeld architectuur IMAP/POP*

Hoofdstuk 4

WWW

Het world wide web is één van de populairste toepassingen van het Internet. Veel mensen maken er dagelijks gebruik van. Het protocol wat hiervoor gebruikt wordt is het http protocol. Versie 1.1 van dit protocol is de meest gebruikte versie en is beschreven in rfc2616[1]. Een aantal belangrijke verbeteringen ten opzichte van versie 1.0 zijn de persistente verbinding en de host header. De host header maakt het mogelijk meerdere webservers te hosten op één webserver / IP-adres, dit worden ook wel virtualhosts genoemd. Door het gebruik van virtualhosts kunnen meerdere virtuele webservers op één server gehost worden; dit heeft tot gevolg dat de server veel te voortduren zou kunnen krijgen. Daarom worden webservers meestal uit meerdere tiers op gebouwd. Over het algemeen ziet dit er schematisch alsvolgt 4.1 uit:



Figuur 4.1: *www flow*

[1]Hypertext Transfer Protocol – HTTP/1.1

De tweede tier hoeft niet altijd aanwezig te zijn, wanneer een webserver alleen statische content heeft, vervalt deze tweede tier. Voorbeelden van applicaties in de tweede tier zijn bijvoorbeeld een search-engine of php.

4.1 Tier 1

Deze tier wordt vaak beschreven als de Presentation layer of de user tier. Dit is de laag welke rechtstreeks communiceert met de client (browser). Wanneer een webserver wordt opgezet heeft men de keuze uit verschillende software pakketten. Op de site van netcraft[2] wordt bijgehouden welke webservers op het Internet gebruikt worden en wat hun markt aandeel is. In tabel 4.1 is weergegeven welke webservers het meest gebruikt werden in december 2003 en januari 2004.

Developer	dec-03	Percent	January 2004	Percent	Change
Apache	31005690	67.43	31040922	67.38	-0.05
Microsoft	9596571	20.87	9675979	21.00	0.13
SunONE	1530372	3.33	1503855	3.26	-0.07
Zeus	749791	1.63	752053	1.63	0.00

Tabel 4.1: *January 2004 Web Server Survey*

Zoals te zien is de apache server veruit de meest gebruikte webserver, 67% van alle webservers zijn apache webservers. Apache is een opensource webserver en voor zowel Unix systemen als Windows beschikbaar.

4.1.1 Scalability

Schaalbaarheid op dit niveau wordt bereikt door horizontaal te schalen. Dit is dus goed schaalbaar. Op het moment dat dit nodig blijkt te zijn, kan er eenvoudig een nieuw systeem naast geplaatst worden. Het verdelen van http requests kan gedaan worden door loadbalancing. Webserver frontends hoeven dus geen dure "zware"machines te zijn wanneer er meerdere parallel geplaatst worden.

4.1.2 Performance

Wanneer een webserver veel virtualhosts bediend kan dit een nadelig effect op de performance van het systeem hebben. Een virtualhost is een virtuele webserver, zo kan een webserver meerdere viruele webservers bedienen en hoeft niet voor iedere virtualhost een aparte webserver te worden ingericht. Zo kan de webserver www.os3.nl ook de website www.snb.science.uva.nl bedienen. Voor iedere virtualhost dienen een aantal vaste parameters gedefinieerd te worden. Er dient geconfigureerd te worden wat de naam van de virtualhost is (bijvoorbeeld www.os3.nl) met eventuele alias namen (bijv os3.nl). Ook dient geconfigureerd te worden waar de logfiles voor deze virtualhost staan en waar de webcontent zich bevindt. Ter illustratie: een eenvoudige configuratie voor een apache virtualhost zou er alsvolgt uit kunnen zien:

```
NameVirtualHost 145.92.24.10
<VirtualHost 145.92.24.10>
```

[2] http://news.netcraft.com/archives/web_server_survey.html

```
    ServerName                  www.os3.nl
    DocumentRoot        /opt/www/www.os3.nl/htdocs
    ScriptAlias  /cgi-bin/  /opt/www/www.os3.nl/cgi-bin
    ErrorLog /var/log/apache/www.os3.nl-error_log
    CustomLog /var/log/apache/www.os3.nl-access_log combined
</VirtualHost>

<VirtualHost 145.92.24.10>
    ServerName                andree.os3.nl
    DocumentRoot        /home/andree/public_html
    ScriptAlias  /cgi-bin/  /home/andree/cgi-bin
    ErrorLog /var/log/apache/andree.os3.nl-error_log
    CustomLog /var/log/apache/andree.os3.nl-access_log combined
</VirtualHost>
```

Wanneer een webserver een aantal honderden virtualhosts heeft, wordt de configuratie-file snel erg groot. Dit heeft tot gevolg dat de webserver veel geheugen verbruikt en langzaam start. Een ander veel voorkomend probleem is dat het systeem te weinig file-descriptors heeft, dit wordt veroorzaakt door dat de webserver voor iedere virtualhost aparte logfiles gebruikt.

Bij de keuze voor een bepaald type webserver, dient rekening gehouden te worden met hoe deze geïmplementeerd is. In de wereld van webservers is grofweg een scheiding te maken tussen twee type webservers namelijk:

Forking servers: dit zijn servers welke voor ieder inkomende http request een server proces "forken". De traditionele webservers zoals de NCSA server en de CERN server waren allemaal forking implementaties. In dit model kunnen er enkele honderden webserver processen bestaan.

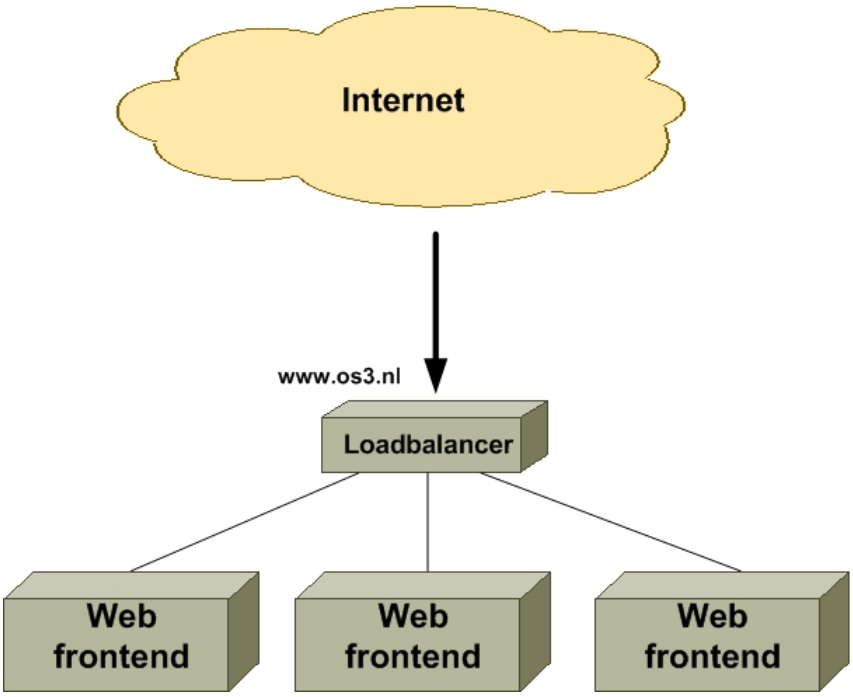
Threading servers: Threads voorzien in een mechanisme voor het managen van meerdere executies binnen één enkel proces. Door gebruik te maken van threads, hoeft niet voor iedere http(s) verbinding een apart proces te worden "ge-forked".

Geen van beide modellen is perfect; beide hebben hun eigen voor en nadelen. Het voordeel van het forken van nieuwe processen voor ieder verzoek, is de eenvoud en betrouwbaarheid. Wanneer er een applicatie fout optreed welke de applicatie doet crashen, heeft dit alleen gevolgen voor dat proces (en die ene client). Alle andere processen worden hier niet door beïnvloed en blijven gewoon draaien. Wanneer dit zelfde gebeurd bij een threaded server zal het hele proces en dus de hele webserver crashen (alle bijbehorende clients zullen hiervan dus last ondervinden). Het voordeel van het threaded model, is dat het een betere performance heeft. Het opstarten van een nieuwe thread binnen een proces kost relatief weinig recources vergeleken met het opstarten van een heel nieuw proces. Ter illustratie: "op een Solaris server verbruikt de functie `thread_create` 90% minder CPU recources dan de functie `fork`". Dit heeft tot gevolg dat een multi-threaded server over het algemeen een groter aantal simultane verzoeken kan afhandelen. Ook het starten en uitvoeren van een nieuwe thread gebeurd veel sneller dan het starten van een nieuw proces, dit verhoogd de "response time"van de webserver en kan zodoende weer meer verzoeken per seconde afhandelen. Er bestaan ook een aantal hybride oplossingen. Zo zijn er bijvoorbeeld de pre-forking webservers. Dit is hoe de Apache (1.3.xx) webserver werkt. In plaats van het starten van een nieuw proces bij een nieuw http verzoek, worden er van te voren al een aantal processen gestart. Op deze manier heeft de

forking overhead geen invloed op de response tijd van de webserver. Er zijn ook implementatie die de forking en threading methode combineren. Deze implementaties starten een aantal multi-threaded processen, om zo het beste van beide methode te bereiken.

4.1.3 Availability en Reliability

Om de beschikbaarheid van de webserver(s) te verhogen worden meerdere servers parallel geschakeld. HTTP verzoeken kunnen op verschillende manieren worden verdeeld over één van de web frontends. Dit kan gedaan worden met round robin dns of door middel van loadbalancers, zie figuur 4.2.



Figuur 4.2: *webservers loadbalancen*

Door dat een loadbalancer kennis heeft van de service welke deze bedient, in dit geval een webserver (http), weet deze wanneer een webfrontend niet juist meer functioneert. Een loadbalancer controleert dit door middel van checks, op het moment dat een check bij één van de frontends faalt, worden er geen verzoeken meer doorgestuurd naar deze frontend. Op deze manier wordt de beschikbaarheid gegarandeerd.

4.1.4 Manageability

Hoe eenvoudig bovenstaande architectuur te beheersen is, is gedeeltelijk afhankelijk van wat voor product er gebruikt wordt. Echter, door het gebruik van een loadbalancer is het mogelijk één frontend tijdelijk uit de "loadbalancing pool"te halen en hier beheer op te doen. Meestal is het na een configuratie wijziging nodig de webserver te herstarten, zodat de wijzigingen actief worden. Wanneer een webserver enkele duizenden virtualhosts heeft, kan het herstarten van een webserver al

snel enkele tientallen minuten duren. Door het gebruik van meerdere frontends en een loadbalancer is dit geen probleem, mits ze niet allemaal tegelijk worden herstart. Configuratie wijzigingen hoeven in principe maar éénmalig gedaan te worden, wanneer alle frontends dezelfde configuratie file gebruiken. Deze file kan bijvoorbeeld door middel van een nfs mount beschikbaar gesteld worden. Centrale configuratie komt de Manageability ten goede. Samenvattend kan geconcludeerd worden dat een dergelijke architectuur relatief eenvoudig te beheren is.

4.1.5 Adaptability

Wanneer op een gegeven moment blijkt dat een bepaalt product niet langer aan de verwachting voldoet, kan er voor gekozen worden een andere webserver te gaan gebruiken. In de architectuur zoals hiervoor geschetst is het relatief eenvoudig om een ander product in te zetten op de eerste tier. Echter wanneer gebruik gemaakt wordt van dynamische content, bijvoorbeeld als ASP en PHP wordt dit wat gecompliceerde.

4.1.6 Security

Omdat de frontends rechtstreeks te benaderen zijn vanaf Internet, dienen deze goed beveiligd te zijn. Dit houdt in dat het systeem up-to-date dient te zien met betrekking tot patches voor zowel het OS als de webserver. Communicatie met de clients kan beveiligd worden door gebruik te maken SSL/TLS, veel websites met transactie of login functionaliteit bieden deze mogelijkheid. Voor het schaalbaarheids aspect, is het belangrijk te beseffen dat door de aard van het SSL protocol, voor iedere virtualhost welke door middel van SSL bereikbaar dient te zien een apart IP-adres nodig is. Tijdens het opzetten van de SSL communicatie, is namelijk nog niet bekend welke virtualhost er wordt opgevraagd, dit komt pas na dat de SSL verbinding is opgezet. Daarom wordt aan de hand van het IP-adres gekeken, welk certificaat gebruikt moet worden. Het is technisch wel mogelijk meerdere SSL virtual hosts op één IP-adres te hosten, maar deze zullen dan altijd van het zelfde certificaat gebruik maken. Of dit een probleem is, hangt af van de gewenste functionaliteit van SSL. De functie van https is eigenlijk tweeledig.

1. Het encrypten van de data welke wordt verstuurd (Confidential + inetgriteit).
2. Zeker weten dat je met de juiste webserver communiceert (authentication).

Deze 2 functies worden met behulp van certificaten bewerkstelligd, in het geval van meerdere SSL-virtualhosts op één IP-adres blijft de encryptie blijft wel functioneren. Echter punt 2 niet, want men weet niet zeker met welke webserver gecommuniceerd wordt. Het certificaat wordt altijd uitgegeven voor één virtualhost, zodoende kan gegarandeerd worden dat je met die betreffende (virtuele) webserver gecommuniceerd wordt. Het is dus niet mogelijk een certificaat voor voor meerdere virtualhosts te gebruiken. Vaak is dit voor websites met minder gevoelige informatie niet zo'n heel groot probleem. Deze gebruiken SSL-virtualhosts vaak voor afgeschermde delen van hun website (htaccess) of webmail, zodat hun password niet "gesniffed"kan worden. Echter voor websites zoals telebankieren is die tweede functionaliteit wel belangrijk. Men wil namelijk wel zeker weten dat men met de website van de bank communiceert en niet met iemand die de site heeft nagemaakt en de client daar op een of andere manier na toe heeft gekregen[3]. Daarvoor wordt het SSL certificaat ook gebruikt, om te bewijzen dat de server wel is wie hij beweert dat hij is.

[3]Men in the middle attack

4.2 Tier 2

Deze tweede tier is niet altijd aanwezig of apart zichtbaar. Dit is afhankelijk van wat de functie van de webserver is. Vaak zijn de eerste en de tweede tier samengevoegd. In de tweede tier is vaak een applicatie ondergebracht, voorbeelden van een dergelijke applicatie zijn:

- Search engine
- webmail applicatie

Het is ook mogelijk dat deze tweede tier verantwoordelijk is voor het uitvoeren of aanroepen van functies door middel van bijvoorbeeld SOAP[4]. Maar ook de software welke verantwoordelijk is voor het genereren van dynamische content kan in deze laag ondergebracht worden, hierbij kan gedacht worden aan PHP[5], ASP[6] CGI[7] en Java. Het is niet altijd mogelijk deze applicaties te scheiden van de eerste tier.

4.2.1 Scalability

Hiervoor geldt in principe het zelfde als op de eerste tier, horizontaal schalen is op dit niveau vaak goed mogelijk. Een alternatieve mogelijkheid is gebruik maken van een "reversed proxy" op de eerste laag en op de tweede laag een webserver met applicatie.

4.2.2 Performance

Performance van deze tweede tier is erg afhankelijk van de gebruikte applicatie. Wanneer hier bijvoorbeeld een search engine gebruikt wordt kan de performance verbeterd worden door juiste indexering.

4.2.3 Availability en Reliability

Van deze applicaties kan de availability en de reliability verbeterd worden door deze in een cluster onder te brengen of te loadbalancen.

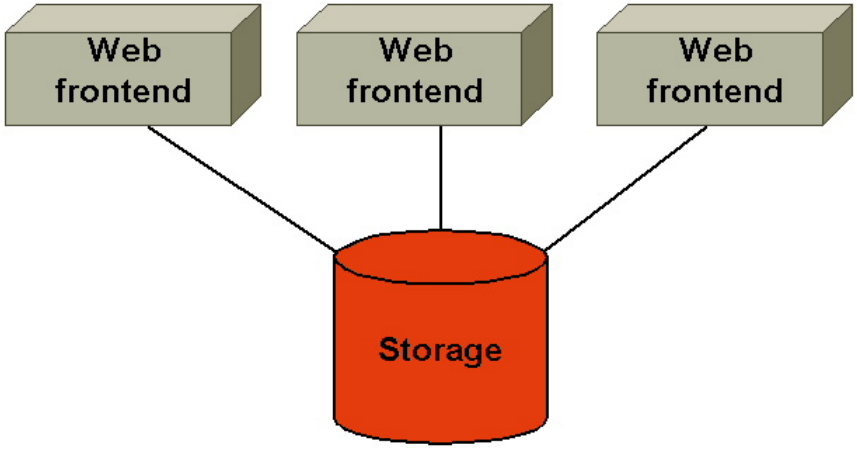
4.2.4 Security

Security speelt een belangrijke rol wanneer op de webserver PHP, CGI of ASP wordt aangeboden. Door het aanbieden van deze diensten wordt de mogelijkheid gegeven tot het uitvoeren van scripts op de webserver. Dit hoeft niet per definitie slecht te zijn, maar hier dient zeker van te voren over worden nagedacht. De door gebruikers gemaakte scripts doen namelijk niet altijd wat er verwacht wordt (of juist wel) met als gevolg dat dit negatieve gevolgen kan hebben voor de performance van de webserver. Bovendien heeft een gebruiker nu toegang tot delen van het filesyteem waar het in principe niets te zoeken heeft. Om deze en andere redenen is het aan te raden om de webserver in een chrooted omgeving te draaien. Dit betekent dat, wanneer de webserver op wat voor manier dan ook "ge-hacked"wordt, niet per definitie het hele systeem compromised is.

[4] Simple Object Access Protocol
[5] PHP: Hypertext Preprocessor
[6] ASP: Active Server Pages
[7] CGI: Common Gateway Interface

4.3 Tier 3

In deze tier vinden we de daadwerkelijke content, zoals html files, cgi-scripts en andere bestanden. Over het algemeen is het aan te raden hiervoor een centrale fileserver voor te gebruiken. Een NFS server is hier een goed voorbeeld van. Bij deze opzet dient iedere frontend webserver een NFS mount te maken naar de nfs server. Zodoende kunnen alle frontend webservers over dezelfde data beschikken. Schematisch ziet dit er alsvolgt uit, zie figuur 4.3:



Figuur 4.3: *web storage*

4.3.1 Scalability

Door de machine welke verantwoordelijk is voor de storage voldoende uit te rusten met storage mogelijkheden, wordt de mogelijkheid voor toekomstige uitbreiding gegarandeerd. Deze derde tier hoeft niet perse een server te zijn welke zijn storage toegankelijk maakt voor andere hosts. Dit kan ook een dedicated storage machine zijn zoals een NAS. Bij de aanschaf van storage dient ten alle tijden rekening gehouden te worden met toekomstige uitbreiding.

4.3.2 Performance

Het filesysteem zal voornamelijk benaderd worden voor read operaties. Hierbij dient dus rekening gehouden te worden met de keuze van hardware. Bij het gebruik van een fileserver over het netwerk (NAS), zoals het Network File System (NFS) of het Common Internet File System (CIF aka SMB) is het verstandig dit over een apart netwerk te laten lopen. Dit zou bijvoorbeeld het backend netwerk kunnen zijn. Zodoende wordt de bereikbaarheid en optimale performance gegarandeerd.

4.3.3 Availability en Reliability

Door de storage uit te voeren met bijvoorbeeld een raid controller of door software raid toe te passen, is het systeem beter gewapend tegen defecte disks. Hierdoor kan de beschikbaarheid van de data beter worden gegarandeerd. Er kan voor gekozen worden de storage onder te brengen bij bijvoorbeeld de mailstorage. Zodoende hoeft er maar één storage array beheerd te worden.

4.3.4 Manageability

Over het algemeen zal er weinig beheer plaatst te hoeven vinden op de fileserver. De taken welke met regelmaat kunnen voorkomen is het aanmaken van nieuwe directories voor nieuwe virtualhosts of het toekennen van de juiste rechten. Deze procedure is bovendien eenvoudig te automatiseren.

4.3.5 Adaptability

Het aanpassen van de tot stand gekomen architectuur is op dit niveau eenvoudig te implementeren. Dit is één van de voordelen van het modulair, het in verschillende lagen, opbouwen van de architectuur. Het is relatief eenvoudig alle data te verhuizen naar een nieuwe storage eenheid. Dit kan een een SAN of een NAS zijn (zie hiervoor het hoofdstuk "Storage").

4.3.6 Security

In principe hoeft de storage alleen beschikbaar te zijn voor de webservers. Rechtstreekse toegang vanaf het Internet naar de fileservers is niet nodig en is dan ook ten zeerste af te raden. Wanneer er een NAS wordt gebruikt, is het belangrijk dat deze goed is beveiligd tegen toegang vanaf andere host dan de webservers. Dit kan door de storage in een apart netwerk te zetten en deze te beveiligen door middel van ACL's[8] of firewall rules. In het geval van direct attached storage speelt dit geen rol.

[8] Access Control Lists, restrictie regels welke worden geïmplementeerd op routers

Hoofdstuk 5

Centrale rol van LDAP

Binnen organisaties kan er behoefte zijn aan het centraal beheren van gegevens. Dit geldt ook voor gebruikers gegevens. Een Directory server leent zich uitstekend voor dit doel en wordt in veel professionele ICT infrastructuren toegepast. Aanvankelijk was het begrip 'Directory Service' vrijwel synoniem met X.500, een standaard op dat gebied. Inmiddels heeft LDAP X.500 zo goed als vervangen als Directory Service protocol.

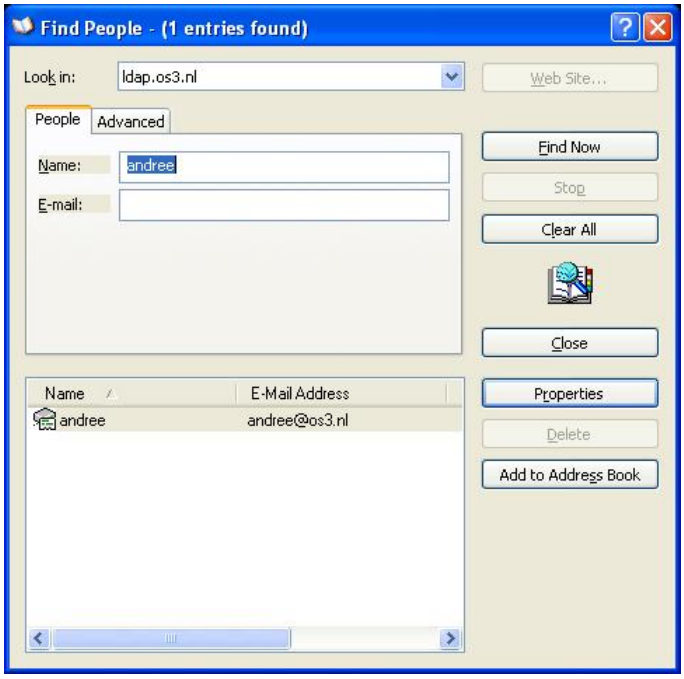
5.1 Centrale gebruikers database

Een LDAP server is in feite een grote database met allerlei gegevens over gebruikers. De volgende gebruikers gegevens worden typisch opgeslagen in een LDAP server:

- Voornaam
- Achternaam
- Telefoonnummer
- Adres
- Emailadres
- PGP sleutelworen
- wachtwoord

Gebruikers kunnen deze database gebruiken als gecentraliseerd adressenboek. Tegenwoordig zijn ook veel email client uitgerust met de mogelijkheid om een LDAP server te gebruiken als adressen boek. In figuur 5.1 is een voorbeeld weergegeven van het gebruik van een LDAP server als adressen boek.

Niet alleen eind gebruikers kunnen gebruik maken van de centrale userdatabase. Ook veel andere servers hebben de mogelijkheid om hiervan gebruik te maken. Zo bieden veel mailservers (smtp, imap4 en pop3) en ftp servers een koppeling met een LDAP server voor authenticatie en authorisatie. Dit kan in grote ICT organisaties veel voordelen bieden. Wanneer servers een koppeling met LDAP niet ondersteunen, zouden deze allemaal een lokale database of password file moeten hebben. Dit is beheers technisch niet wenselijk, het wordt helemaal ingewikkeld om deze password files dan ook nog gesynchroniseerd te houden.



Figuur 5.1: *LDAP-adressenboek*

5.2 Centrale configuratie

Een LDAP server kan naast opslag van gebruikers gegevens ook voor de opslag van configuratie files gebruikt worden. In de vorige hoofdstukken is al aan de orde gekomen dat er voor één functie vaak meerdere servers (frontends) ingezet kunnen worden. Het zou handig zijn als deze allemaal dezelfde configuratie file zouden gebruiken. Dit maakt het beheer een stuk eenvoudiger omdat de configuratie files allemaal centraal staan opgeslagen. Doordat deze configuratie files centraal zijn opgeslagen wordt het maken van wijzigingen een stuk eenvoudiger en zeker wanneer dit geautomatiseerd gebeurd, bijvoorbeeld door een provisioning syteem.

5.3 Single point of failure

Door deze diensten, zoals configuratie files, te centraliseren wordt wel een bepaald risico genomen. Wanneer er zich een probleem voor doet met de LDAP server, kan er geen authenticatie en authorisatie meer plaats vinden, servers kunnen geen configuratie files meer inlezen, mailservers kunnen geen email meer afleveren, etc. Het is dus belangrijk om de LDAP omgeving met zorg op te zetten. Hoe dit gedaan kan worden en waar rekening mee gehouden dient te worden is beschreven in het volgende hoofdstuk.

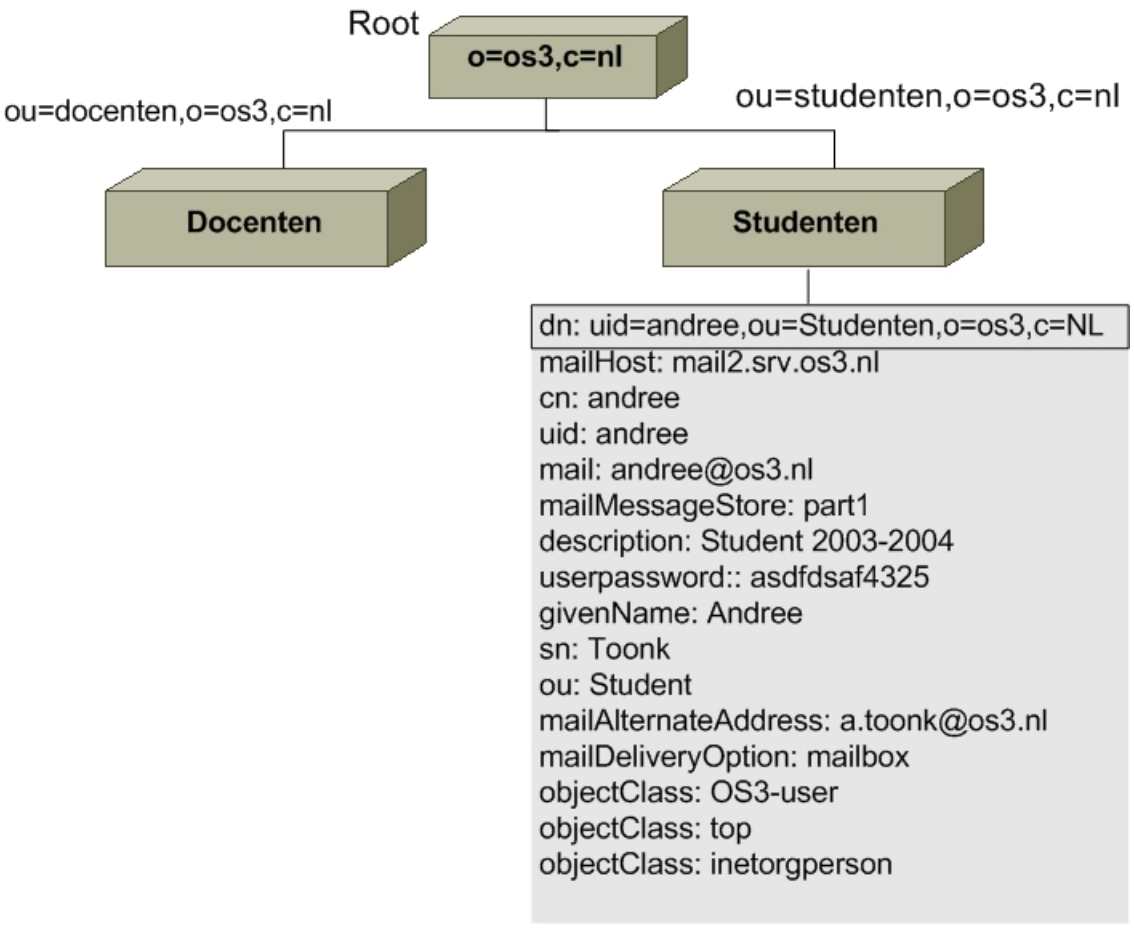
Hoofdstuk 6

LDAP

LDAP, het Lightweight Directory Access Protocol, biedt de mogelijkheid om e-mail adressen, telefoongegevens of welke grote structuur gegevens dan ook snel te kunnen raadplegen vanuit diverse programma's. Veel e-mail clients ondersteunen LDAP standaard als een methode om e-mail adressen op te zoeken. In het systeembeheer wordt LDAP vaak gebruikt om gebruikers gegevens centraal op te slaan zodat deze gegevens te raadplegen zijn vanaf meerdere servers. Er zijn een aantal verschillen tussen LDAP en een normale database. De informatie in een LDAP directory is zo opgeslagen dat gegevens snel te vinden zijn, vele malen sneller dan in een "normale"database. Het is dus niet bedoeld om op regelmatige basis gegevens weg te schrijven. In principe is een LDAP server een database server welke te benaderen is via het LDAP protocol. Een Ldap server is geoptimaliseerd voor zoek en lees acties. De meest recente versie van het LDAP protocol is versie 3, dit is beschreven in RFC2251.

LDAP is een directory server en heeft over het algemeen een duidelijke boom structuur. Hoe deze structuur wordt opgebouwd is afhankelijk van degene welke dit ontwerpt. Een voorbeeld structuur is weergegeven in figuur 6.1.

In het figuur is te zien dat de root wordt beschreven als "o=os3,c=nl". Dit staat voor organisation=os3 en Country=nl. Daaronder is een onderverdeling gemaakt in organizational units, namelijk: studenten en docenten. Daaronder zijn de users beschreven, in LDAP wordt dit gedaan met behulp van zogenaamde LDIF's, ("LDAP Data Interchange Format"). Een voorbeeld LDIF is hieronder afgebeeld.



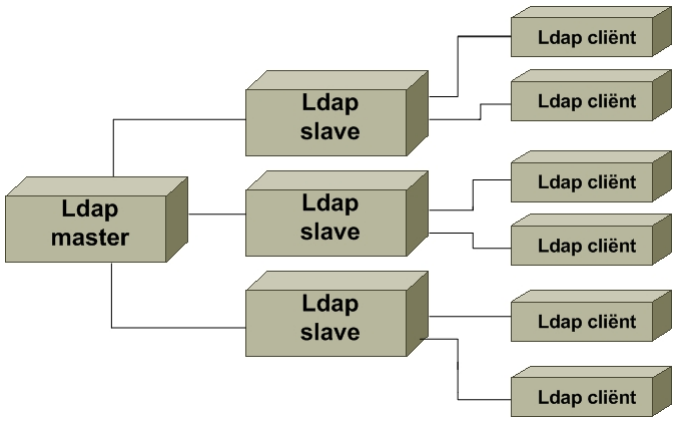
Figuur 6.1: *LDAP-directory*

dn: uid=andree,ou=Studenten,o=os3,c=NL
mailHost: mail2.srv.os3.nl
cn: andree
uid: andree
mail: andree@os3.nl
mailMessageStore: part1
description: Student 2003-2004
title: Student
givenName: Andree
sn: Toonk
ou: Student
mailAlternateAddress: a.toonk@os3.nl
objectClass: OS3-user
objectClass: top
objectClass: inetorgperson

Een dergelijke LDIF is een beschrijving van de persoon in de LDAP database. Door middel van LDIF bestanden is het mogelijk wijzigingen aan te brengen in de LDAP database, ook de uitvoer van searches zijn in LDIF formaat.

6.1 topology

Door dat LDAP een belangrijke rol kan spelen in de huidige ICT infrastructuren, is het belangrijk deze goed op te zetten. Gelukkig helpt LDAP ons hierbij, door het master slave principe te gebruiken. Het is met LDAP, net als bij DNS, mogelijk één of meerdere masters te gebruiken welke het recht hebben om LDAP entries te wijzigen. Daarnaast zijn er de LDAP slaves welke alleen gebruikt kunnen worden om te lezen. Het is belangrijk een juiste topology te kiezen, zodat bottlenecks, welke veroorzaakt worden door overbelaste LDAP servers, niet kunnen voor komen. Een voorbeeld toplology is weergegeven in afbeelding 6.2.



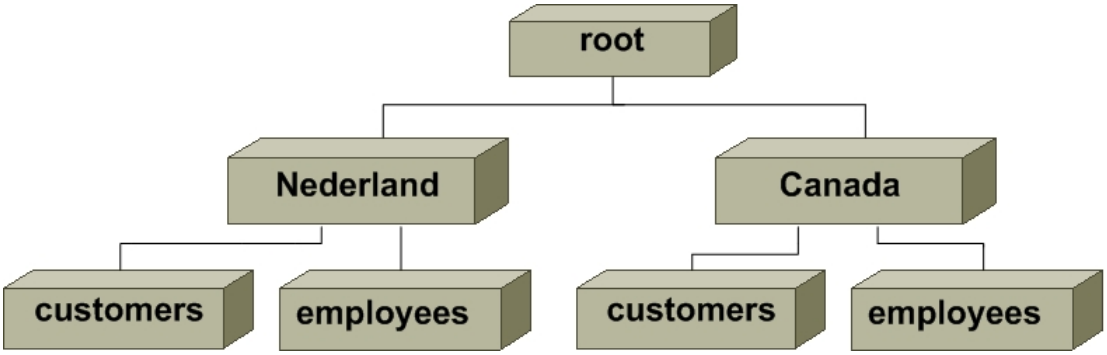
Figuur 6.2: *LDAP-topology*

In deze topologie is er één master en een drietal slaves. Er is altijd minimaal één master nodig, zodat er wijzigingen gemaakt kunnen worden in de LDAP database. Veel voorkomende wijzigingen zijn bijvoorbeeld veranderingen van het userpassword attribuut.

In afbeelding 6.2 is te zien dat de LDAP clients alleen kunnen communiceren met de slaves. Dit is gedaan omdat zodoende de master wordt ontzien en zich alleen hoeft bezig te houden met het bijwerken van wijzigingen (schrijf acties). Wanneer een gebruiker zijn of haar password wil wijzigen zal deze dat, via een daarvoor ingerichte plaats (bijvoorbeeld de webmail applicatie of provisioning syteem), doorgeven aan één van de slaves. Omdat een slave geen schrijfrechten heeft en alleen maar in de database kan lezen, zal deze het verzoek doorsturen naar de master. De master krijgt het verzoek binnen en zal, mits deze gebruiker rechten heeft dit bepaalde attribuut te wijzigen, het attribuut wijzigen. Deze wijzigingen zal daarna worden gerepliceerd naar de slaves, deze worden ook wel replicatie servers genoemd.

6.2 Scalability

Schaalbaarheid van de LDAP servers kan eenvoudig worden bereikt door het bijplaatsen van LDAP slaves en eventueel een tweede master. Het bijplaatsen van nieuwe slaves is in principe niet aan een maximum gebonden, er kan dus eenvoudig horizontaal geschaald worden. Wanneer LDAP word gebruikt binnen, bijvoorbeeld een multinational, kan het verstandig zijn om voor ieder land/werelddeel een aparte LDAP server in te zetten. In het voorbeeld van figuur 6.3 is een onderscheid gemaakt



Figuur 6.3: *LDAP-directory2*

tussen werknemers en klanten in Canada en Nederland. Door gebruik te maken van "referals"hoeft de LDAP database in Nederland niet alle data van gebruikers in Canada te hebben. Ook wijzigingen hoeven nu niet naar elkaar gerepliceerd te worden.

6.3 Performance

De performance van een individueel systeem kan eenvoudig worden verbeterd door de LDAP server juist te laten indexeren, zodat het zoeken in de database sneller gaat. Ook is het verstandig een onderscheid aan te brengen in LDAP slaves en masters. Het is aan te raden de master(s) alleen te gebruiken voor schrijf acties en de slaves alleen voor lezen. Zo kunnen beide type servers voor hun eigen specifieke taak geoptimaliseerd worden.

6.4 Availability en Reliability

De beschikbaarheid en de betrouwbaarheid van de LDAP service kan worden verhoogd door zowel de LDAP slaves als de masters te loadbalancen. In plaats van loadbalancing kan er ook gebruik gemaakt worden van round robin DNS, maar het gebruik van loadbalancers verhoogd de availability net iets meer omdat deze in staat is te anticiperen op wijzigingen. Round robin DNS daarentegen is veel statischer en is niet in staat onmiddellijk te reageren op outages van een LDAP server. Door gebruik te maken van een multi-master opzet is het mogelijk de LDAP masters goed te loadbalancen. LDAP-modifys op Master A worden gerepliceerd naar zowel de Slaves als Master B.

6.5 Security

Omdat LDAP een belangrijk onderdeel is van veel van de huidige ICT infrastructuren, is het belangrijk dit goed te beveiligen. Dit is mogelijk door de masters en de slaves te scheiden door ze in een twee verschillende netwerken onder te brengen. De clients hebben alleen toegang nodig tot de slaves, het is dus verstandig de masters helemaal af te schermen en alleen toegankelijk te maken voor de slaves. De meeste LDAP server implementatie bieden extra mogelijkheden om de data goed af te schermen. Zo beschikken alle gangbare LDAP servers over de mogelijkheid om AccesLists te implementeren. Hiermee kan worden gedefinieerd wie welke entries wel of niet mag lezen of wie welke entries wel of niet mag wijzigen. LDAP versie 3 biedt de mogelijkheid te werken met beveiligde verbindingen door gebruik te maken van SSL/TLS. Dit zorgt ervoor dat alle communicatie encrypt wordt verstuurd. Het is aan te raden de LDAP servers onderling gebruik te laten maken van deze mogelijkheid en waar mogelijk ook de communicatie tussen clients en servers.

6.6 Manageability

Voor het beheren van een LDAP database zijn verschillende mogelijkheden. Ontwikkelaars van commerciële LDAP servers leveren daarbij vaak een GUI aan waarmee de LDAP directory beheerd kan worden. Een bekende opensource implementatie van LDAP is de OpenLdap Server[1]. Deze server is, net als iedere andere server, te managen door middel van commandline tools. Verder is het handig om door de directory te kunnen bladeren, door dat LDAP een openstandaard is, zijn hiervoor een groot aantal tools[2] beschikbaar welke voor iedere LDAP server te gebruiken zijn. Een backup maken van de LDAP database kan op twee manieren

- Een backup maken van de complete database
- Een LDIF export maken.

Optie 2 is de meest flexibele manier, op deze manier kan men product onafhankelijk werken. De inhoud van de LDAP database kan zo bijvoorbeeld geëxporteerd worden naar een andere LDAP server. Een tweede voordeel van een LDIF export, is dat de beheerder felxibeler kan omgaan met deze data. Het is hiermee bijvoorbeeld mogelijk slechts een gedeelte van de data terug te zetten. De backup van de complete database, is met name handig wanneer de hele database corrupt is geraakt.

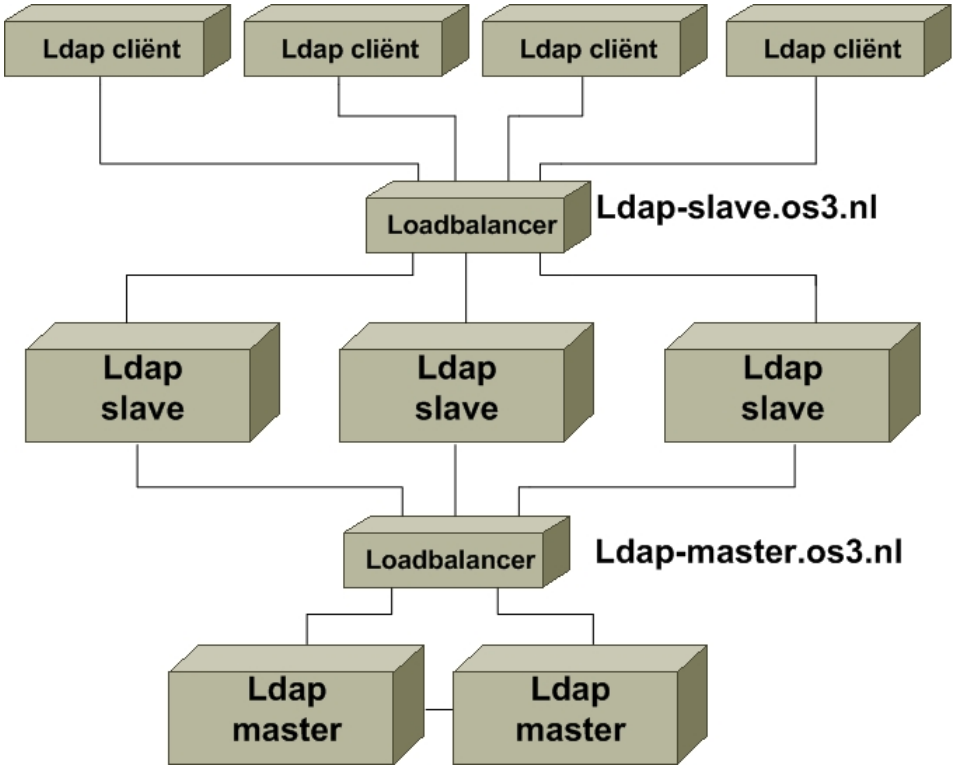
[1] http://www.openLDAP.org/
[2] Java LDAP Browser/Editor: http://www.iit.edu/ gawojar/LDAP/

6.7 Adaptability

Wanneer ooit besloten wordt een andere software te gaan gebruiken, voor de LDAP server moet dit als het goed is niet uitmaken en mogelijk zijn. Dit omdat LDAP een open standaard is. Helaas is dit niet altijd het geval. Zo kan de 'Active Directory' server van Microsoft niet zomaar vervangen worden door een andere LDAP server. Dit ondanks dat Microsoft pretendeert de LDAP standaarden te gebruiken. Bovendien dient er bij een eventuele migratie, van bijvoorbeeld de Iplanet Directory server naar OpenLDAP rekening gehouden moeten worden met features die product afhankelijk zijn. Een voorbeeld hiervan zijn de ACLs. Deze zijn op de Directory server van SUN veel preciezer te configureren dan dat dit in de OpenLDAP server mogelijk is.

6.8 Voorbeeld architectuur

Rekening houdend met bovenstaande punten, zou een voorbeeld architectuur er alsvolgt uit kunnen zien, zie figuur 6.4. Alle clients communiceren alleen met de slaves. De masters zijn ook niet te benaderen door de clients, deze zijn alleen bereikbaar door de slaves. Availability en Reliability wordt bereikt door meerdere slaves te load-balancen. Ook de masters worden ge-loadbalanced, zo wordt ook de performance van de LDAP service als geheel verbeterd. Wanneer er na figuur 6.4 gekeken wordt, zijn ook hier weer 3 tiers in te ontdekken.



Figuur 6.4: *LDAP-architectuur*

Hoofdstuk 7

Storage

Storage is een onderdeel van de data-tier welke de opslag van data verzorgd. Dataopslag is een belangrijk en kostbaar deel van de ICT-infrastructuur. Het is de repository van een organisatie. In de veel gevallen kan een organisatie niet functioneren als toegang tot de informatie wegvalt. Dataopslag is de spil waar alles om draait. Te zien is dat de hoeveelheid data welke een organisatie opslaat jaarlijks in veel gevallen verdubbeld. Vooral rekencentra vereisen veel opslagcapaciteit. De exponentiële groei vereist een gedegen aanpak van dataopslag binnen de ICT-infrastructuur.

7.1 Centrale opzet van storage

Om de explosieve groei aan te kunnen, moet de dataopslag schaalbaar zijn met voldoende performance. Dit laatste is een belangrijke factor omdat de bottleneck bij servers meestal trage disktoegang is. Performanceverbetering van dataopslag zal tot een totale performanceverbetering binnen een infrastructuur leiden. In grote omgevingen heeft het voordelen om data op een centrale plaats op te slaan. Het op een willekeurige plaats centraal opslaan van data vereist een ontwerp welke van invloed is op de totale infrastructuur. Het is belangrijk om bij het opzetten van een infrastructuur al in een vroeg stadium na te denken over de manier van dataopslag en de daarbij toegepaste kostbare technologieën om zo voor zowel de servers als de storage efficiënt te kunnen investeren.

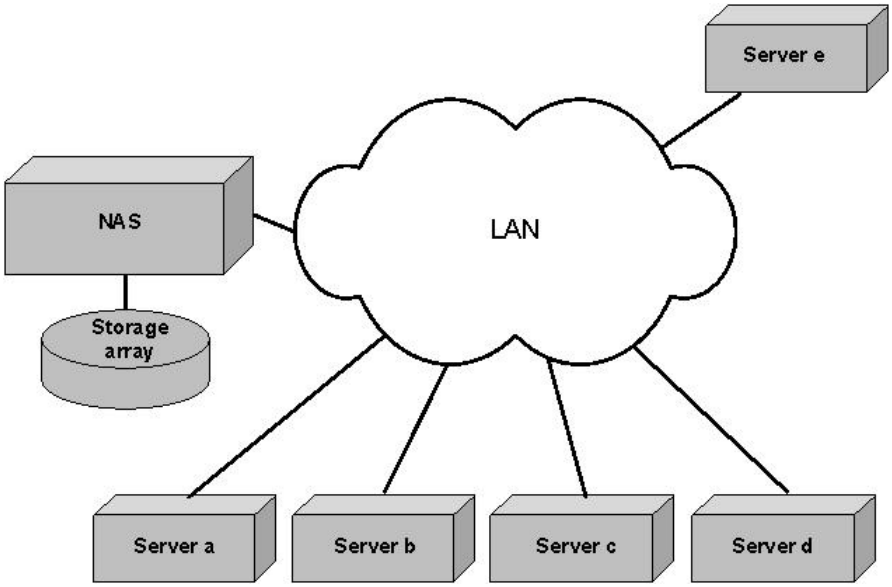
Het centraal plaatsen van dataopslag heeft de volgende voordelen:

- Schaalbaar door eenvoudige uitbreiding van de capaciteit
- Het dynamisch toewijzen van een hoeveelheid schijfruimte aan de verschillende servers
- Topologische flexibiliteit door onafhankelijkheid van een fysieke locatie
- Redundant op te zetten door spreiding over de verschillende locaties
- Scheiding aanschaf van dataopslag en de servercapaciteit
- Hogere beschikbaarheid te realiseren door concentratie van dataopslag
- Centraal managen van data en uniform beheer van de totale storage
- Centraal en efficiënt gebruik van backup devices

DAS (directed attached storage) was de traditionele manier van dataopslag welke in grote omgevingen niet voldoet door het ontbreken van de voordelen welke zojuist genoemd zijn. Met het centraal opslaan van data wordt storage losgekoppeld van de servers. Dit houdt in dat individuele schijven of disk-arrays met IDE, SCSI of fibre channel arbitrated loop (FCAL) niet meer direct met een server zijn verbonden zoals dat bij DAS het geval is. De centralisatie is met twee verschillende technologien te realiseren:

- Network attached storage (NAS)
- Storage area network (SAN)

Voor het aanbieden van data maakt SAN gebruik van block-gebaseerde protocollen zoals SCSI terwijl NAS gebruik maakt van filesharingprotollen zoals NFS. Door dit verschil zijn SAN en NAS storage oplossingen met ieder hun eigen toepassingsgebied.

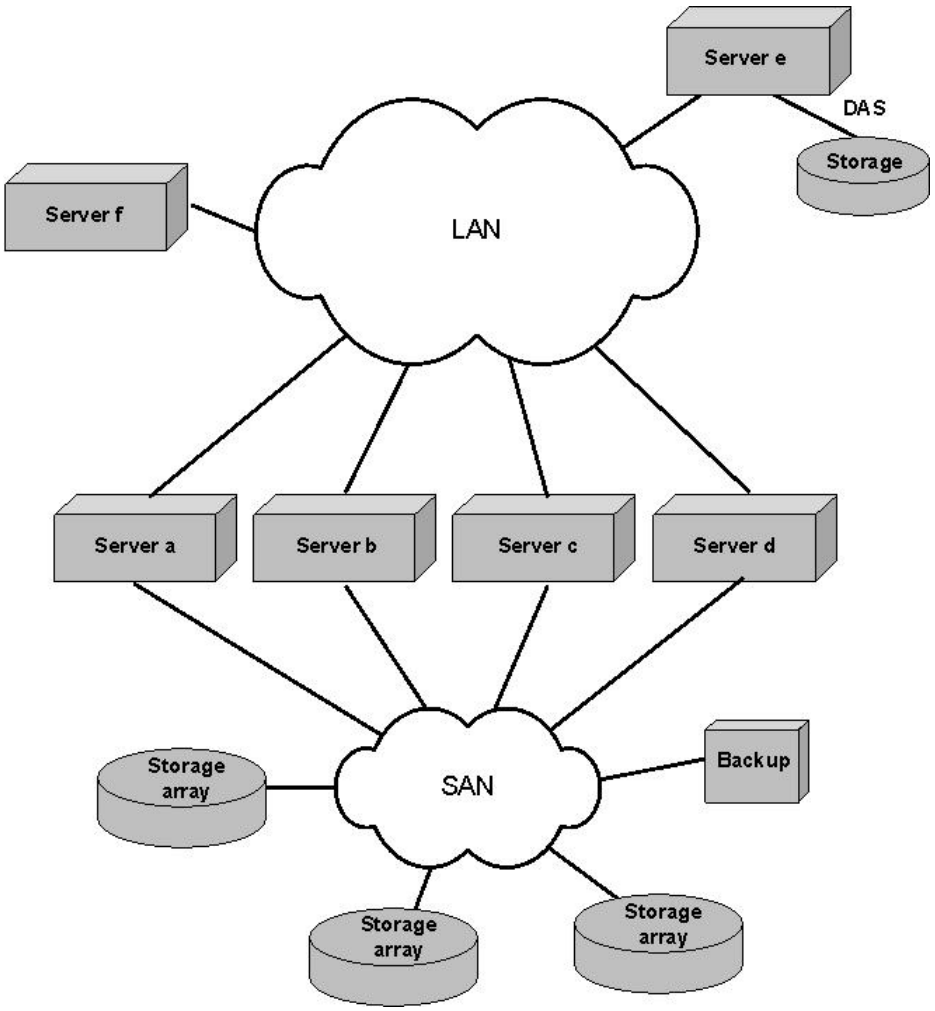


Figuur 7.1: *Network attached Storage*

7.2 NAS

Met NAS wordt de dataopslag door middel van een netwerk-protocol over een LAN toegankelijk gemaakt (zie figuur 7.1). Een voorbeeld hiervan is de NFS file-server. Een NAS-server kan een dedicated in hardware uitgevoerd device zijn of een services welke door een server wordt geleverd. De dedicated NAS-server neemt de vertaling van filesysteem-I/O naar block level disk I/O voor zijn rekening. Het voordeel is dat de file-servers van deze taak worden ontlast. De applicatie-servers worden echter nog wel belast met het door de TCP/IP-stack sturen van gegevens, een niet te onderschatten belasting. NAS maakt data voor meerdere servers toegankelijk. Data kan dus worden gedeeld. Dit is een voordeel van NAS. Het delen brengt echter wel file-locking problemen met zich mee. Hoe de data daadwerkelijk door de NAS-server opgeslagen wordt, is voor de servers welke gebruik maken van NAS geheel onzichtbaar. Doordat data via een LAN toegankelijk wordt

gemaakt, kan deze data centraal in een netwerk geplaatst worden al dan niet verdeeld over meerdere NAS-servers. Het nadeel van NAS is de lage performance. De doorvoersnelheid van data is vooral van kleine files veel lager dan wanneer er van een direct gekoppelde disk gebruik wordt gemaakt .



Figuur 7.2: *Storage Attached Network*

7.3 SAN

Een SAN is een eigen netwerk 'achter' de servers dat storage devices en de servers met elkaar verbindt (zie figuur 7.2). Een SAN kan gezien worden als een pool met storage devices welke beschikbaar worden gesteld aan de servers. Een server kan zo dataopslag betrekken van fysiek verschillende storage devices. In tegenstelling tot NAS, belast SAN door het gebruik van dat eigen netwerk het bestaande LAN niet. Hoewel het mogelijk is om eenvoudig een SAN te realiseren met uitsluitend SCSI-hardware, wordt bij wat grotere SAN-oplossingen veelal gebruik gemaakt van speciaal voor SAN ontworpen protocollen zoals Fibre Channel (FC). De Fibre Channel architectuur wordt het meest toegepast. Deze architectuur kan met fiber worden uitgevoerd. Voor kortere afstanden wordt meestal koper gebruikt. Het Fibre Channel (FC) protocol is een datatransportprotocol zoals ether-

net dat voor een LAN is. Dit protocol kan SCSI commando's vervoeren binnen de switched Fibre Channel omgeving. Later is Fibre Channel Arbitrated Loop (FCAL) als nieuwer protocol binnen de Fibre Channel architectuur ontwikkeld dat erg populair is geworden. Hiermee kunnen point-to-point verbindingen gemaakt worden maar ook is een loop van in serie geschakelde point-to-point verbindingen mogelijk. Dit protocol ondersteunt geen centrale switching functie en is shared.

Een SAN bestaat uit hardware componenten die een netwerkstructuur vormt waarbij er gebruik wordt gemaakt van switches voor een sterstructuur of van hubs voor een logische ringstructuur. Op een SAN kunnen verschillende soorten storage devices aangesloten worden. Er zijn harddisks, RAID controllers en Tape Libraries die over een FCAL interface beschikken. De devices kunnen met een dubbele interface uitgevoerd zijn en bieden zo redundantie op device niveau. Verder bestaan er bridges welke bestaande SCSI devices aan een Fibre Channel netwerk kunnen koppelen. Servers worden met een host bus adapter kaart al dan niet via een media-convertor op een SAN-hub, switch of in een loop aangesloten. Voor Fibre Channel switches bedraagt de snelheid momenteel 1 of 2 Gbit per seconde. Dit houdt in dat de disks zelf nog steeds de vertragende factor zijn en niet de SAN.

Fiber maakt de storage onafhankelijk van de fysieke locatie. Zo kan de dataopslag zich op 10 kilometer afstand van de servers bevinden. Met SAN kan per server een hoeveelheid diskruimte van willekeurige storage devices toegewezen worden. De servers en de daarop draaiende applicaties zien geen verschil tussen een interne disk of een externe SAN-disk. De performance is gelijk. Het is met SAN nog niet mogelijk om meerdere servers dezelfde data te laten delen zoals dit wel het geval is bij NAS. In de toekomst zal met de opkomst van zowel SAN-filesystemen als IP-gebaseerde SAN's dit wel mogelijk worden, het verschil tussen SAN en NAS zal daarmee wegvallen.

De technologie van SAN is sterk in ontwikkeling. San zal verder als technologie niet diepgaand besproken worden. Er zal vooral bekeken worden hoe SAN in een ICT-infrastructuur ingepast kan worden waarbij de Fibre Channel architectuur als uitgangspunt zal worden genomen.

7.4 Scalability

De data-tier is door het makkelijk uitbreiden van dataopslag in een SAN erg schaalbaar geworden. Dit uitbreiden van dataopslag bestaat uit het bijplaatsen van storage dat op de SAN wordt aangesloten. Door de onafhankelijkheid van de locatie kan extra strorage op elke willekeurige locatie worden geplaatst. De applicatie-tier is schaalbaar door het dynamisch kunnen toekennen van diskruimte. Ook is clustering van de applicatie-tier eenvoudiger te realiseren. Bij clustering vereist een enkele fysieke server meerdere partities voor de verschillende services welke geclusterd worden. Veelal zijn deze partities door middel van mirroring ook op een fysiek ander storage device geplaatst. Alle servers binnen een cluster moeten toegang hebben tot deze partities. Met SAN wordt de flexibiliteit geboden dat de clusters schaalbaar maakt.

7.5 Performance

Performance van storage kan worden verhoogd voor RAID te gebruiken. Dit wordt voor de performance in hardware met RAID-controllers gerealiseerd. Met RAID-5 wordt de data verspreid over meerdere harddisks opgeslagen. Lezen en schrijven gebeurt parallel over meerdere disks waardoor

de snelheid wordt verhoogd. Een logische partitie op een RAID-5 cluster met 10 kleine harddisks performt beter dan een cluster met 4 grote harddisks terwijl de logische partitie dezelfde opslagcapaciteit kunnen hebben. Er zal hier een evenwicht gezocht moeten worden tussen de kosten en de snelheidswinst welke met gebruik van meerdere harddisks voor een bepaalde opslagcapaciteit te behalen is. Verder kan de performance verhoogd worden door diskcaching toe te passen. De disks kunnen uitgerust worden met bijvoorbeeld 1 Gbyte cache geheugen. Hierbij is het wel belangrijk dat bij stroomuitval de data in de cache alsnog naar disk kan geschreven worden om dataverlies te voorkomen.

7.6 Availability en reliability

Storage moet redundant uitgevoerd worden. Data wordt op harddisks opgeslagen. Harddisks bevatten mechanische onderdelen welke slijten. Hoewel de kwaliteit van harddisks steeds beter wordt, kunnen ze onverwachts uitvallen. Om het falen van deze hardware op te vangen, wordt gebruik gemaakt van RAID. RAID-5 is de meest gebruikte RAID-oplossing waarbij data over meerdere disks wordt verdeeld. Een disk kan uitvallen waarna de andere disks de uitgevallen disk opvangt. Behalve de performancevermindering merkt de server hier niets van. In een storage array kan een extra hot standby disk geplaatst worden welke de uitgevallen disk meteen vervangt. Met RAID-5 wordt deze lege disk vervolgens weer van data voorzien. Zo kan de performancevermindering beperkt blijven. Om de beschikbaarheid verder te verhogen, kan naast RAID ook mirroring toegepast worden. Met mirroring op verschillende fysieke storage devices, kan een enkel device zonder problemen in het geheel uitvallen. Ook kan het zinvol zijn om mirroring over verschillende fysieke locaties toe te passen wat met fiber mogelijk is. Bij brand in de storage ruimte of bij aanslagen is beschikbaarheid zo nog steeds gegarandeerd.

7.7 Manageability

Storage is in een centrale opstelling overzichtelijker te beheren dan wanneer dit direct aan de verschillende servers is gekoppeld. Het beheer van SAN is echter complex. Er moet veel kennis aanwezig zijn om SAN te kunnen configureren en onderhouden. Het onderhoud van een stabiele situatie zal niet veel beheerslast geven. Het configureren en wijzigen echter kunnen tot instabiele situaties leiden. De beschikbaarheid van storage is zeer belangrijk, problemen met SAN dienen zeer snel opgelost te worden dus beheerders moeten direct en adequaat kunnen handelen. Door identieke storagehardware te gebruiken, is kennis van slechts een beperkt type devices nodig dat in veelvoud wordt gebruikt en kunnen compabiliteitsproblemen worden vermeden.

7.8 Adaptability

Omdat in het verleden met de DAS-systemen geen noodzaak bestond voor koppelingen tussen verschillende aanbieders, heeft elke leverancier zijn eigen variatie op de standaarden geïmplementeerd. Dit heeft tot gevolg dat iedere vendor zijn eigen implementatie van een SAN op de markt bracht. Met de opmars van FCAL worden standaarden strakker door de vendors gehanteerd. Het koppelen is wel mogelijk maar de kans op onvoorspelbare problemen is aanwezig. Het achterhalen van deze problemen kan erg veel tijd kosten waarbij in het ergste geval de storage niet beschikbaar is. Vendors geven support op eigen hardware. Meestal wordt er voor een SAN een supportcontract

afgesloten en beperkt men zich tot een enkele vendor. Zolang vendors afwijken van standaarden zal de flexibiliteit in aanpassingen van SAN een probleem blijven.

7.9 Security

Data moet veilig opgeslagen worden. Alleen de betreffende servers mogen toegang hebben tot de data. SAN's kennen nog geen goede beveiliging zoals authenticatie en encrytie. Vooral met interlokale verbindingen moet hier voorzichtig mee omgegaan worden. Doordat storage een onderdeel is van de data-tier, is alleen toegang mogelijk door de applicatie-servers. Gebruikers hebben daardoor niet direct toegang tot de data. Verder moet data altijd terug te halen zijn. Backup is zeer belangrijk. Storage kan wel redundant uitgevoerd worden, maar data kan ook door een foutieve handeling van een beheerder verloren gaan. Het snel recoveren van data is belangrijk. Backup-tapes mogen niet gaan zwerven of zoek raken. Door de backup centraal te gaan regelen met strenge procedures, wordt voorkomen dat backup-tapes makkelijk verduisterd kunnen worden.

Hoofdstuk 8

Provisioning

Binnen een grote organisatie kan het aanbieden van ICT services een zware belasting vormen op het beheer. Vooral bij een groot aantal gebruikers welke van bepaalde services gebruik maken is de ondersteuning hiervan niet te onderschatten. Er kunnen binnen het beheer erg tijdrovende processen aanwezig zijn welke handmatig worden uitgevoerd. Hierbij is de kans op het maken van fouten vrij groot en is dit werk kostbaar. Het is zinvol om dit soort processen te inventariseren en te kijken toe hoever het efficiënt is om deze te automatiseren. Provisioning verzorgt de geautomatiseerde processen en wordt als gereedschap gebruikt voor het aanbieden van ICT services. Het kan als een belangrijke beheerstechnologie gezien worden.

Voorbeelden van provisioning zijn:

- Het beheren van useraccounts
- Het beheren van virtuele webservers
- Het beheren van mailboxes
- Het beheren van toegang en rechten

Processen kunnen met provisioning eenvoudiger en minder arbeidsintensief worden voor beheerders, helpdeskmedewerkers en eventueel voor de gewone gebruikers. Provisioning zorgt voor afscherming van complexiteit zodat de gebruiker geen kennis behoeft van moeilijke en systeemspecifieke configuraties. Veelal wordt provisioning in de praktijk met een webinterface uitgevoerd. Hierdoor is met een webbrowser vanaf elke willekeurige locatie op eenvoudige wijze toegang mogelijk tot het provisioning-systeem.

8.1 Scalability

Met provisioning wordt het mogelijk om verschillende toegangsrechten en niveau's aan gebruikers toe te kennen zodat verantwoordelijkheden en functies aan bepaalde personen kunnen worden toegewezen. Dit is vooral zinvol in grote omgevingen met veel beheerders en helpdeskmedewerkers welke zich op verschillende plaatsen in een organisatie bevinden. Ook wordt het mogelijk om centraal aangeboden ICT-services op eenvoudige wijze decentraal aan te kunnen passen aan de wensen van de beheerders en gebruikers. Dit maakt de services vanuit het beheer gezien schaalbaar. Zo kan

bijvoorbeeld een persoon binnen een bepaalde afdeling worden belast met het beheer van mail-accounts. Het aanmaken, beheren en verwijderen van mail-boxen kan door iemand gebeuren welke geen kennis bezit van het complexe mailsystemen.

8.2 Performance

Meerdere mensen kunnen tegelijkertijd gebruik maken van het provisioning-systeem. Hierdoor neemt de capaciteit van het doorvoeren van wijzigingen toe. Men is niet van één persoon of één helpdesk afhankelijk. Ook kan met provisioning zelfs real-time gewerkt gaan worden. In plaats van verzoeken te deponeren bij een helpdesk welke deze met een zekere frequentie behandelden, worden verzoeken meteen gerealiseerd. De productiviteit neemt hierdoor toe.

8.3 Availability en reliability

Omdat de daadwerkelijke configuratiewijzigingen door een provisioning-systeem worden uitgevoerd is er minder kans dat er fouten ontstaan. Het provisioning-systeem dwingt de gebruiker om gegevens op een bepaalde manier in te voeren. De informatie wordt gecontroleerd alvorens deze wordt geaccepteerd. Zo worden menselijke fouten voorkomen. Hierdoor is de kans kleiner geworden dat een service door een foutieve configuratie zal falen. Een provisioning-systeem moet robuust opgezet worden. Bij uitval van dit systeem is geen enkele wijziging meer uit te voeren. Een organisatie is hiervan dus afhankelijk. Afhankelijk van de implementatie zal het systeem in de meeste gevallen uit een gescheiden database- en applicatie-server bestaan. Deze kunnen dubbel uitgevoerd worden waarbij de database-server als mirror meedraait. Als een van de servers uitvalt, kan eventueel handmatig de standby servers actief gemaakt worden.

8.4 Security

Het toekennen van verschillende functionaliteiten aan de gebruikers is een vorm van beveiliging. Door de infrastructuur zo op te zetten dat configuratiewijzigingen uitsluitend via een provisioning-systeem kunnen gebeuren, worden ook beheerders gedwongen dit systeem te gebruiken. Gedetacheerde beheerders welke voor een beperkte tijd werkzaam zijn binnen een organisatie, kunnen toegang tot het provisioning-systeem verleend worden zonder verregaande rechten welke misbruikt kunnen worden. Verder kunnen events in een provisioning-systeem gelogd worden zodat altijd te achterhalen is wie een bepaalde handeling heeft uitgevoerd.

8.5 Manageability

Het beheer van een provisioning-systeem is niet intensief. Veelal bestaat het syteem uit een database- en een applicatie-server. Het beheer zal vooral bestaan uit het maken en aanpassen van scripts waarmee het provisioning-systeem de daadwerkelijke configuraties op de verschillende servers uitvoert.

8.6 Adaptabililty

Een provisioning-systeem is een gereedschap. In de praktijk vinden er continue wijzigingen en uitbreidingen plaats. De infrastuctuur welke veranderd en de wensen van gebruikers vereisen een provisioning-systeem dat dynamisch is. Aanpassingen moeten daarom eenvoudig te realiseren zijn. Een goed provisioning-systeem heeft twee transparante interfaces. Een naar de gebruikers en een naar de systemen. De interface naar de gebruiker kan in de vorm van een webservice met een WSDL worden aangeboden. In de praktijk wordt de client meestal door dezelfde softwareontwikkelaars als het provisioning-systeem ontwikkeld zodat deze interface niet zichtbaar geïntegreerd is. De interface naar de systemen zorgt voor het afhandelen van de commando's om de configuratie uit te voeren. Deze roepen de scripts aan van de specifieke systemen. Een specifiek systeem moet gewijzigd kunnen worden zonder dat dit zware veranderingen teweeg brengt voor het provisioning-systeem. Alleen wijzigingen van de scripts door de beheerders zou voldoende moeten zijn.

Hoofdstuk 9

Security

Security is een hot-item tegenwoordig en dat moet ook! Langzamerhand begin een hoop organisaties in te zien dat het loont hierin te investeren. In iedere organisatie dient de beveiliging gebaseerd te zijn op beleid. Het is belangrijk van te voren goed na te denken over zaken die met de beveiliging van het netwerk en de servers te maken hebben.

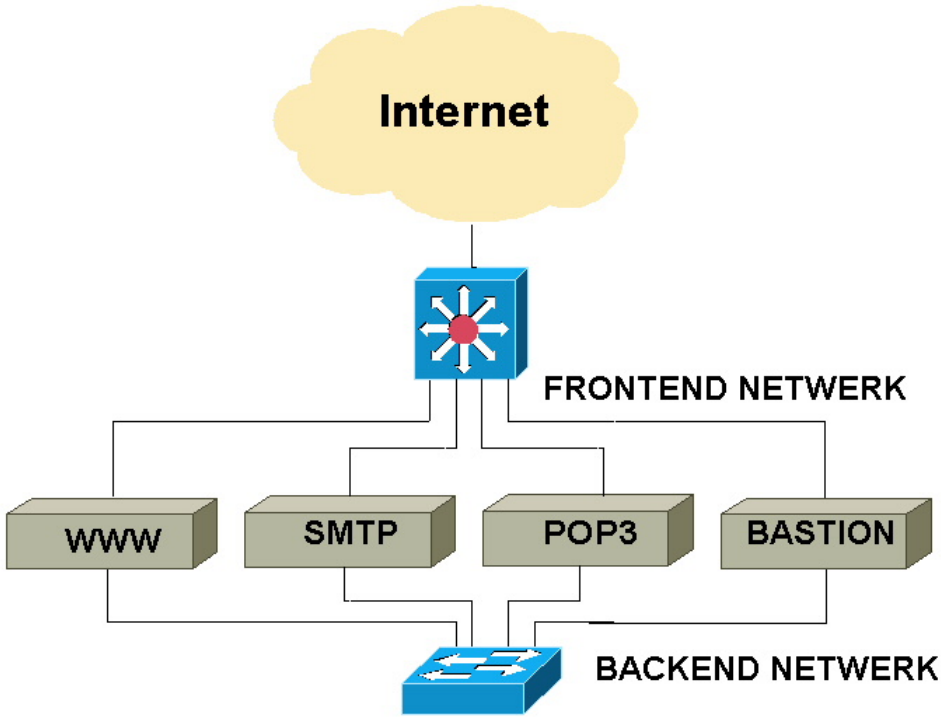
Er kan hierbij een onderscheid gemaakt worden tussen 3 verschillende niveaus.

- Security op netwerk niveau
- Security op systeem niveau
- Security op gebruikers niveau

9.1 Security op netwerk niveau

Door de toegang tot het netwerk te beveiligen met acceSSLists of een stafull firewall/packetfilter is het mogelijk een gedegen basis te leggen voor een veilige omgeving. Door alleen verkeer toe te laten tot het netwerk wat expliciet is toegestaan, weet de beheerder wat deze voor verkeer kan verwachten. Hiermee wordt het security risk al enorm gereduceerd.
Het is verstandig om de infrastructuur in twee netwerken te onder te verdelen, een frontend netwerk en een backend netwerk. Het frontend netwerk wordt gebruikt om netwerk access voor de aangeboden diensten te faciliteren. Hiervoor moeten publieke IP-adressen worden gebruikt. Systemen dienen vervolgens een tweede interface te hebben voor de connectiviteit met het backend netwerk. Dit netwerk wordt gebruikt voor toegang voor beheerders en communicatie tussen de hosts. De publieke services, zoals bijvoorbeeld een webserver, hoeven alleen op het frontend IP-adres te luisteren voor verzoeken. Voor beheers specifieke services zoals een ssh daemon of windows terminal service, is het van belang dat deze alleen op het backend netwerk luisteren voor verzoeken.

Voor toegang tot het backend netwerk kan een host gebruikt worden welke toegang heeft tot het backend netwerk en welke toegankelijk is vanaf het Internet. In Figuur 9.1 is deze host aangeduid met de naam "bastion". Toegang tot de bastion host dient alleen toegestaan te worden vanaf een aantal specifieke netwerken, bijvoorbeeld die van de beheerspartner. Alleen vanaf de bastionhost is verdere toegang tot alle servers mogelijk (via het backend netwerk). Bij de beveiliging op netwerk niveau, hoort ook een "Network Intrusion Detection Sytem"Een dergelijk syteem stelt de beheerders in staat een goed inzicht te krijgen in wat voor potentiële aanvallen op de systemen zijn uitgevoerd.



Figuur 9.1: *netwerk beveiliging*

Het is echter wel belangrijk te beseffen dat het "kwaad" dan vaak al geschied is, des al niet te min kan een IDS de beheerder waardevolle informatie verschaffen.

9.2 Security op systeem niveau

Security op dit niveau kan op verschillende manieren worden verbeterd. Een goede regel is alleen software pakketten te installeren welke ook gebruikt zullen gaan worden. Zodoende wordt de hoeveelheid software pakketten welke onderhouden moeten worden verkleint. Verder is het belangrijk de juiste patches te installeren voor zowel het operating system als de extra software. De patches voor het Operating system zijn vaak eenvoudig te vinden op de site van de ontwikkelaar, bijvoorbeeld www.microsoft.com of www.sun.com. Echter het is bijna ondoenlijk om alle sites van de ontwikkelaars van de andere sofware bij te houden. Daarom is het aan te raden een aantal relevante mailinglists bij te houden. De volgende lists/sites zijn aan te raden:

- Full-Disclosure[1]
- Computer Emergency Response Team[2]
- Securityfocus[3]

[1] http://lists.netsys.com/pipermail/full-disclosure/
[2] http://www.cert.org/
[3] http://www.securityfocus.com/

Deze lists en sites worden vulnerabilities van allerlei soorten software aangekondigd en bediscussieerd. Er zijn ook tools beschikbaar welke de diverse servers kunnen testen op vulnerabilities (http://www.nessus.org/). Dergelijke tools kunnen nuttig zijn bij het controleren van de security van het netwerk en de hosts. Ook het 'loggen' van allerlei informatie speelt een belangrijke rol. De meeste servers loggen allerlei informatie in lokale logfiles. Dit kunnen logs zijn van het operating system, alsmede van de diverse applicaties welke op de server draaien. Het is belangrijk deze logfiles regelmatig te controleren. Zo kunnen vaak vroegtijdig (security) problemen opgemerkt worden. Een centrale server waar naar toe de servers loggen, maakt het controleren van de logfiles een stuk minder tijdrovend.

9.3 Security op gebruikers niveau

Het is belangrijk alleen user accounts aan te maken voor degene die dat echt nodig hebben. Voor server systemen zijn dit normaal gesproken accounts welke nodig zijn voor beheers specifieke taken. Er moet bedachtzaam worden omgegaan met de hoeveelheid rechten die worden toegekend aan een user. Voor sommige taken zijn wellicht extra rechten nodig, onder *nix systemen kan hiervoor sudo worden gebruikt. Uiteraard is het ook een belangrijk de eindgebruikers goed voor te lichten, hierbij kan gedacht worden aan het aanbieden van anti-virus software.

Hoofdstuk 10

Monitoring

Wanneer een server park zorgvuldig is opgezet en er is rekening gehouden met zaken als:

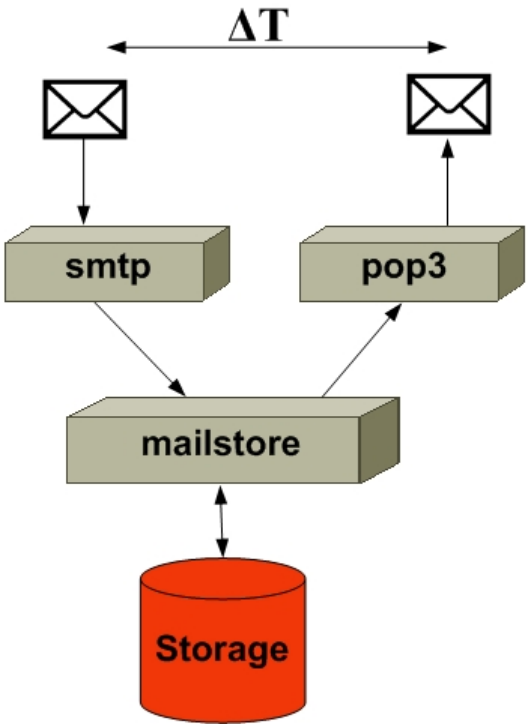
- Scalability
- Performance
- Availability en Reliability
- Security
- Manageability
- Adaptability

is er een goede basis gelegd voor een degelijke ICT infrastructuur. Maar het stopt hier niet! Het is belangrijk alle diensten te monitoren zodat fouten en outages vroegtijdig opgemerkt kunnen worden en niet pas opgemerkt worden na aanleiding van klachten van eindgebruikers. Er zijn verschillende manieren om diensten te monitoren, dit is afhankelijk van wat er gemonitoord dient te worden en tot welk detail.

Een relatief eenvoudige manier van monitoren van diensten, zijn de functionele checks. Dit houdt in dat er wordt gecontroleerd of dat de diensten die de server zou moeten bieden ook nog geboden worden. Voor een webserver geld bijvoorbeeld dat deze web pagina's moet kunnen serveren. Wanneer er wat verder wordt gegaan met het implementeren van checks, zou gekeken kunnen worden naar hoe lang het duurt voor dat de dienst haar werk heeft gedaan. Een voorbeeld hiervan is de mailflow, zie figuur 10.1

Hier wordt de tijd gemeten (ΔT) tussen het versturen van een email en het weer ophalen met het POP3 protocol. Op deze manier wordt niet alleen de Availability getest maar ook de response tijden wat eigenlijk de Performance representeert.

Er kan pas echt van pro-aktief beheer gesproken worden op het moment dat ook andere, aan de service verwante zaken gemonitoord worden. Hierbij kan gedacht worden aan de load van het systeem, raid errors of de hoeveelheid vrij ruimte op de hard disks. Deze zaken kunnen de beschikbaarheid van de diensten verstoren en wanneer deze niet gemonitoord worden, wordt dit pas opgemerkt wanneer de diensten er last van ondervinden.



Figuur 10.1: *mail-monitoring*

Er bestaan reeds kant en klare monitorings pakketten welke vaak goed functioneren. Het kan echter voorkomen dat deze niet altijd over de features beschikken welke gewenst zijn. Er kan dus ook voor gekozen worden een eigen product te ontwikkelen. In dat geval is het verstandig te werken met het SNMP protocol [1]. Dit protocol is speciaal ontworpen voor het managen en beheren van objecten via het netwerk en kan de ontwikkelaar van een dergelijk product veel werk uit handen nemen.

Uiteraard dienen er ook events te worden gekoppeld aan de monitoring. Voorbeelden van mogelijke events zijn

- sms bericht
- pieper bericht (pager)
- email
- webpage

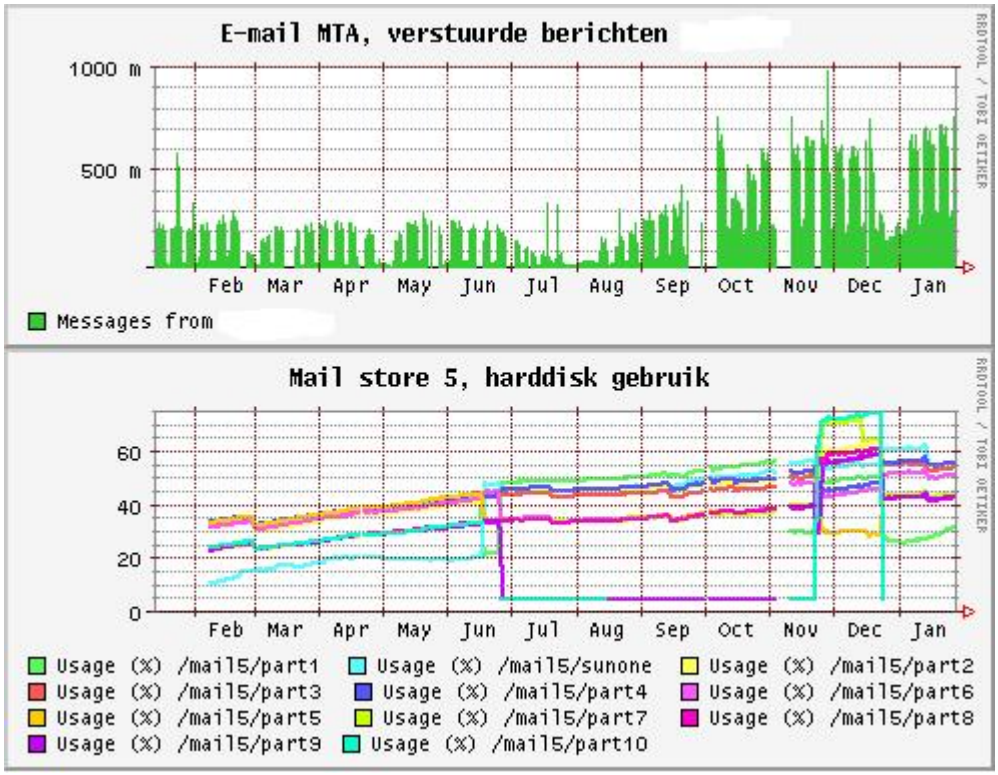
De eerste 2 berichten zijn push events, de eigenaar van de mobiele telefoon of pager wordt er op geattendeerd dat er wat mis is. In het geval van een event door middel van een email of webpage, spreken we van een "pull event".

Het kan ook erg nuttig zijn om de gemeten waarden uit te zetten in grafieken. Op deze manier kunnen trends worden waargenomen en kan beter pro-actief beheerd worden. Belangrijke trends kunnen zijn:

[1] Simple Network Management Protocol

- Hoeveelheid email berichten die per tijdseenheid worden verstuurd
- De tijd die een email er over doet om verzonden en weer ontvangen te worden (ΔT van de mailflow)
- Aantal LDAP queries per tijdseenheid.
- De load van een server

Door deze waarden grafisch uit te zetten kunnen trends waargenomen worden en kan er op tijd actie ondernomen worden. Een voorbeeld hiervan is te zien in figuur 10.2, hier zijn het aantal email berichten welke door één MTA zijn verstuurd grafische weergegeven. In het figuur daaronder is het harddisk gebruik van de mailstores weergegeven.



Figuur 10.2: *mail statistieken*

Hoofdstuk 11

Conclusie

Tot slot kan geconcludeerd worden dat diensten zoveel mogelijk modulair opgezet dienen te worden om belangrijke aspecten zoals schaalbaarheid en beschikbaarheid te kunnen garanderen. Door deze diensten op te delen in een model, zoals het in dit document gebruikte 3-tier model, zijn deze aspecten eenvoudiger te begrijpen en te realiseren. Een mooi voorbeeld hiervan is de mail infrastructuur. In veel van de grote ICT infrastructuren is dit lagen model terug te vinden. Wanneer de infrastructuur van een dienst in verschillende lagen is opgezet, is met name 'adaptablity' van de infrastructuur hoog. Op deze manier hoeven wijzigingen op een laag, geen invloed te hebben op een andere laag. Een praktijk voorbeeld hiervan is door op de derde tier (data) de direct attached storage te vervangen door een SAN. SAN oplossingen zullen de komende paar jaren een enorme groei gaan doormaken. Dit is met name de danken doordat een SAN goed schaalbaar is. NAS implementaties zullen blijven bestaan en wellicht gebouwd worden op de SAN infrastructuur. Wanneer gebruik gemaakt wordt van een NAS techniek zoals NFS, dienen van te voren alle voor en nadelen goed afgewogen te worden. NFS is bijvoorbeeld niet erg geschikt voor mail infrastructuren. Dit komt met name door de performance van NFS en de locking problemen. Als 'fileserver' of storage voor webpages is NFS daarentegen erg geschikt. Het gebruik van een directory server zoals LDAP voor centrale authentcatie en authorisatie is aan te bevelen in grote ICT infrastructuren. Omdat LDAP is gebaseerd op een open standaard zijn diensten zoals web, mail, news en andere servers ook geschikt voor LDAP authenticatie. Het is belangrijk de LDAP infrastructuur redundant op te zetten. Juist doordat deze dienst zo belangrijk is kan het ook een single point of failure zijn. Wanneer de LDAP service binnen een organisatie uitvalt, kan dit tot gevolg hebben dat veel diensten zoals email en news en FTP onbruikbaar worden. Wanneer de infrastructuur is opgezet dienst deze gemonitoord en beheerd te worden. Om het beheer te eenvoudigen is het aan te raden een "provisioning syteem"te gebruiken. Met een dergelijk systeem kunnen eind-gebruikers, helpdeskmedewerkers en systeembeheerders éénvoudig wijzigingen maken. Door de bestaande diensten te monitoren, worden problemen vroegtijdig ontdekt en kunnen potentiële capaciteitsproblemen in een vroeg stadium opgemerkt worden.

Bronnen

Request for Comments (RFC)

RFC 2251
Lightweight Directory Access Protocol (v3)

RFC 1777
Lightweight Directory Access Protocol

RFC 1487
X.500 Lightweight Directory Access Protocol

RFC 821
Simple Mail Transfer Protocol

RFC 2821
Simple Mail Transfer Protocol

RFC 2822
Internet Message Format

RFC 2487
SMTP Service Extension for Secure SMTP over TLS

RFC 2617
HTTP Authentication: Basic and Digest Access Authentication

RFC 2616
Hypertext Transfer Protocol – HTTP/1.1

RFC 1939
POP3

Websites

Netcraft
http://news.netcraft.com/

Apache
http://httpd.apache.org/

OpenLdap
http://www.openldap.org/

Postfix
http://www.postfix.org/

SUN Blueprints
http://www.sun.com/solutions/blueprints/

Classical Internet Applications and Operating Systems (CIAOS)
http://www.os3.nl/CIAOS/

The Evolutionary Systems- and Network Administration (ESNA)
http://www.os3.nl/ESNA/

Benchmarking mbox versus maildir
http://www.courier-mta.org/mbox-vs-maildir/

Maildir over NFS
http://www.qmail.org/man/man5/maildir.html

Using maildir format
http://cr.yp.to/proto/maildir.html

Mailbox storage technology
http://www.washington.edu/imap/documentation/formats.txt.html

High Capacity Email
http://www.vergenet.net/linux/mail_farm/html/

IMAP Aggregator
http://asg.web.cmu.edu/cyrus/ag.html

Linux Virtual Server
http://www.linux-vs.org/

Mail Messaging Multiplexor
http://enterprise.netscape.com/docs/messaging/60/admin/mmp.htm

SAN Tutorial
http://www.dothill.com/tutorial/html_tutorial/topic01.htm

Storage Area Networking (SAN)
http://www.cs.utk.edu/ gupta/SAN.doc

Boeken

Sun BLUEPRINTS: Sun ONE Messaging Server
ISBN: 013145496X
Auteur: Dave Pickens

Sun BLUEPRINTS: Designing ISP Architectures
ISBN: 0130454966
Auteur: John V. Nguyen

Configuration and Capacity Planning for Solaris Servers
ISBN: 0133499529
Auteur: Brian L. Wong

Understanding and Deploying Ldap Directory Services
ISBN: 067232316
Auteurs: Tim Howes, Timothy A. Howes, Mark C. Smith, Gordon S. Good

Interviews

Mederwerkers van ZXfactory:
- Jeroen de Meijer
- Bas Toonk

Lijst van figuren

Lijst van tabellen