Host-based vs dedicated routers

Ofwel: Zebra vs Cisco

Iljitsch van Beijnum

Er bestaat een lange traditie van het gebruik van gespecialiseerde apparatuur voor het routeren van IP pakketten. En een nog langere traditie van "gewone" computers die als routers functioneren. Wat is beter? Meer specifiek: is een Cisco router beter dan een Unix machine die Zebra routingsoftware draait, of is het misschien andersom?

Waarom routers?

Het internet hangt met routers aan elkaar. Waarom eigenlijk? Voor iemand die minder bekend is met het OSI-model en de strikte uitsplitsing van functionaliteit naar verschillende lagen, ligt zoiets niet voor de hand. Wanneer een ethernet op één locatie verbonden moet worden met een ethernet op een andere locatie via een niet-ethernet verbinding zoals een huurlijn, is het blijkbaar de taak van de routers om te zorgen voor de nodige conversies om de ethernet-pakketten over de lange-afstandsverbinding te versturen. Maar ook binnen een bedrijfsnetwerk dat alleen uit ethernet bestaat, of in een datacenter waarbij de verbinding naar een internet service provider een gewone ethernetkabel is, staan routers. Wat is het dat deze routers doen wat niet door een etherswitch gedaan kan worden?

Routeren

Het essentiële verschil tussen een router en een switch is dat een router veel dieper in de pakketten kijkt. Een router kan daardoor het pakket, net als een switch, direct bij de juiste machine afleveren, maar hij kan het pakket ook op basis van alleen het netwerk-deel van het IP adres alleen een eind in de goede richting sturen, in de hoop dat routers verderop meer specifieke informatie hebben. Hierdoor is routeren aanzienlijk beter schaalbaar dan switchen. Een erg grote switch kan misschien 10.000 individuele MAC adressen aan. IP routers die BGP draaien moeten alle mogelijke bestemmingen over de hele wereld in hun routingtabellen hebben. Dit is aanzienlijk meer: er zijn op dit moment wereldwijd ruim 110.000 onafhankelijke IP adresreeksen in gebruik. En ieder van die adresreeksen staat weer voor tientallen tot duizenden op het internet aangesloten systemen.

Routingprotocollen

Een switch doet na het opstarten geen moeilijke dingen, maar gaat gelijk aan het werk. Als er een pakket binnenkomt waarvan de switch geen idee heeft waar het heen moet (omdat het bestemmings MAC adres nog onbekend is) kopieert hij het pakket naar alle poorten. Wel onthoudt de switch het bron MAC adres, gekoppeld aan de poort waarop het pakket ontvangen werd. Als er dan een pakket geadresseerd aan dit MAC adres binnenkomt, kan het direct naar de goede poort gestuurd worden. Iedere switch doet dit voor zich zonder coordinatie met andere switches.

Een router daarentegen, herkent lokale bestemmingen aan het feit dat het netwerkdeel van het IP adres in een pakket overeenkomt met een van zijn eigen IP adressen. Door middel van routingprotocollen wisselen routers deze informatie met elkaar uit om ook pakketten voor niet-lokale bestemmingen door te kunnen sturen. Routingprotocollen zijn er in soorten en maten: simpele interne routing protocollen zoals RIP, en complexe, zoals OSPF en IS-IS, worden binnen het netwerk van een enkele organisatie gebruikt. Voor het uitwisselen van routinginformatie tussen netwerken van verschillende organisaties is er een extern protocol: BGP versie 4. Wanneer een verbinding uitvalt, reageren routingprotocollen door het IP verkeer via de overblijvende verbinding te sturen. In het geval van BGP kan dit net zo makkelijk een lijn naar een andere ISP zijn als een tweede lijn naar dezelfde ISP of een interne kabel.

Filteren

Een minder princieel, maar erg praktisch verschil is dat routers kunnen filteren op alle mogelijke IP-informatie, terwijl pure switches zelden over genoeg kennis van het IP protocol beschikken om deze taak uit te voeren. Mocht het dus nodig zijn om IP-informatie te filteren, dan volstaat een switch meestal niet, maar is minstens een router nodig. Vaak is een firewall beter, omdat die speciaal voor dit soort werk gemaakt is. Maar de functionaliteit van een (application layer) firewall met alles erop en eraan is lang niet altijd nodig: routers kunnen prima uit de voeten met zaken als anti-spoofing filters die voorkomen dat gebruikers pakketten met valse IP adressen de wereld in kunnen sturen, of filters die de reeksen voor privé-gebruik (10.0.0.0/8, 172.16.0.0/12 en 192.168.0.0/16) tegenhouden.

Soorten routers

Er zijn ruwweg vier soorten routers: dedicated software-based, multi-layer switches, hardware-based en host-based routers.

Dedicated software-based routers

Dit type routers bestaat uit een apparaat dat weinig verschilt van een gewone computer, maar dan zonder de onderdelen die geen bijdrage leveren aan het routeren, zoals een toetsenbord, beeldscherm of (hard) disks. De router beschikt over een CPU, RAM, netwerk interfaces (of slots hiervoor) en flash geheugen om software in op te slaan. Het OS is speciaal toegesneden op het efficiënt routeren en filteren van pakketten, en op het draaien van routingprotocollen.

Als er een pakket binnenkomt krijgt de CPU een interrupt. De CPU controleert het pakket, haalt het door inkomende filters, zoekt de bestemming op in de routingtabel of Forwarding Information Base (FIB), past uitgaande filters toe en zet het pakket in de queue van het uitgaande interface. Daarna keert de CPU terug naar zijn oorspronkelijke taak. Hoe meer pakketten er binnenkomen, hoe hoger de CPU-belasting en hoe trager alles wordt. Ingewikkelde filters versterken dit effect. Software-based routers zijn over het algemeen niet in staat tot wire speed forwarding: de CPU kan het maximale aantal pakketten per seconde dat de interfaces gezamelijk kunnen genereren niet bijhouden.

Voorbeelden van dit type routers: de Cisco 2500, 2600, 3600, 4500, 7200 en 7500 modellen. Ook allerlei "domestic gateways" en kabel/ADSL routertjes vallen in deze categorie, hoewel hun functionaliteit moeilijk te vergelijken is met een "echte" router.

Multilayer switches

Een multilayer switch bestaat uit een combinatie van een switch en een router. Meestal zijn ze gecombineerd in dezelfde behuizing. Een multilayer switch combineert de intelligentie van een router met de performance van een switch door alleen het eerste pakket van iedere "flow" door de router te laten behandelen, en alle volgende pakketten te switchen. Als het eerste pakket van bijvoorbeeld een TCP sessie de filters mag passeren, dan geldt dit voor alle andere pakketten die tot deze sessie of flow behoren ook. De switch herkent flows aan het interface waarop de pakketten binnenkomen, de bron- en bestemmingsadressen (zowel het MAC als IP adres), het protocol (TCP, UDP of anders) en de TCP of UDP poortnummers. Multilayer switches en andere flow-gebaseerde systemen zijn niet voor alle toepassingen even geschikt: wanneer het aantal nieuwe flows per seconde erg hoog is gaat de performance omlaag. Met name binnen netwerken van grote of zelfs middelgrote service providers kan dit gebeuren omdat WWW-verkeer zeer veel nieuwe flows opzet. Het opvragen van een HTML pagina met 20 plaatjes levert immers 21 verschillende TCP sessies op. Als het aantal flows binnen redelijke grenzen blijft kunnen multilayer switches over het algemeen met wire speed routeren/switchen.

Tegenwoordig zijn er niet zoveel "pure" multilayer switches. Veel switches die als "layer 2, 3 en 4" verkocht worden hebben hardware aan boord die ieder pakket apart kan routeren en/of filteren.

Dedicated hardware-based routers

De snelste routers die tegenwoordig beschikbaar zijn hebben speciale hardware (ASICs) aan boord voor het forwarden en filteren van IP pakketten. Die hardware maakt het mogelijk om voor ieder individueel pakket routing- en filtertabellen door te zoeken en op basis van wat daarin gevonden wordt een beslissing te nemen over wat er met het pakket moet gebeuren. Gewoonlijk is de speciale hardware zodanig gedimensioneerd dat routeren met wire speed geen probleem is.

Voorbeelden van dit type routers zijn alle Juniper modellen en de Cisco 12000 serie.

Host-based routers

Tegenwoordig hebben nagenoeg alle operating systems TCP/IP support ingebouwd. De extra code om ook te kunnen routeren is uitermate simpel als de code om als host IP te runnen al beschikbaar is: eigenlijk nauwelijks meer dan het kortsluiten van de routines voor het ontvangen en versturen van IP pakketten. Het kost dan ook weinig moeite om van een gemiddeld Unix systeem een router te maken:

```
# sysctl -w net.inet.ip.forwarding=1
net.inet.ip.forwarding: 0 -> 1
```

Maar alleen het daadwerkelijke routeren (forwarden) van pakketten heeft maar beperkt nut. Dit is eigenlijk alleen noodzakelijk op het moment dat het om twee verschillende typen verbindingen of subnets gaat, bijvoorbeeld in het geval van PPP en ethernet. Routeren-zonder-wat tussen twee ethernetsegmenten heeft weinig zin: switchen is sneller en goedkoper. Routeren tussen twee ethernetten gebeurt in praktijk dus alleen wanneer er filtering of dynamische routering met gebruik van een routingprotocol nodig is.

Unix-varianten zoals Linux en FreeBSD beschikken over uitgebreide IP filtermogelijkheden in de kernel. Wanneer deze ingezet worden raakt de router-functionaliteit meestal wat op de achtergrond en noemen we zo'n machine meestal een firewall. (Hoewel het laatste woord nog niet gesproken is over wat nu wel en niet "firewall" mag heten.)

Forwarden en filteren moet voor ieder pakket apart gebeuren en is daarom uit performance-overwegingen in de kernel geïmplementeerd. Routingprotcollen daarentegen draaien als gewone processen in "userland". Unix systemen beschikken over het algemeen over routed, een daemon die het RIP protocol implementeert. Het voordeel van RIP is dat het niet of nauwelijks enige configuratie behoeft en dus makkelijk in het gebruik is. Dit is gelijk ook een van de belangrijke beperkingen: het is een vrij "dom" protocol dat moeilijk op een ingewikkelder netwerk toe te snijden is. Daarnaast is het verre van efficiënt: RIP genereert veel broadcasts (iedere 30 seconden de volledige routingtabel) en het duurt lang voor wijzigingen in de netwerktopologie hun weg vinden naar de routingtabellen van alle RIP-routers.

De Gated en GNU Zebra software implementeren slimmere protocollen: OSPF voor interne routering en BGP voor extern gebruik.

Performance

De forwarding-performance van een router hangt van drie dingen af: de snelheid van de CPU of andere hardware die het werk moet doen, het "switching path" en de groote van de routingtabellen en filters.

Cisco switching paths

Cisco routers beschikken over verschillende "switching paths", zoals process switching, fast switching, optimum switching en Cisco express forwarding (CEF). Bij process switching worden inkomende pakketten in een buffer geplaatst en moet een gewoon proces ze verwerken aan de hand van de "gewone" (niet-geoptimaliseerde) routingtabel. Dit is traag. De overige switching paths voeren het forwarden in interrupt mode uit, en maken niet langer gebruik van de gewone routingtabel, maar van een geoptimaliseerde "route cache". Bij fast switching is dit een tree (binaire boom). Fast switching route cache entries worden opgebouwd tijdens process switching, de route cache vormt dus over het algemeen niet een volledige afspiegeling van de gewone routingtabel. Optimum switching gebruikt een iets ander soort boomstructuur: een 256-way tree, waarin een individueel IP adres in 4 in plaats van 32 stappen te vinden is.

Fast en optimum switching zijn geoptimaliseerd voor snel zoeken, niet voor een efficiënte opslag van grote hoeveelheden routing-informatie. CEF daarentegen is goed in allebei en gebruikt een 256-way trie structuur, waarbij de interface en MAC-rewrite informatie niet in de nodes van de boom opgeslagen worden, maar in een aparte datastructuur, waar de boom alleen naar verwijst. Nodes kunnen ook terugverwijzen naar andere nodes, waarmee het mogelijk is recursiviteit rechtstreeks in de CEF tabel op te lossen. In tegenstellen tot de fast en optimum switching route cache, wordt de CEF tabel niet tijdens process switching opgebouwd, maar is er een apart proces dat de CEF tabel synchroon houdt met de gewone routingtabel. Strikt genomen is de CEF tabel dus geen cache. Zo'n volledige kopie van de routingtabel voor het forwarden van pakketten wordt vaak een Forwarding Information Base (FIB) genoemd. In software-routers is

CEF het snelste switching path, en hardware-routers maken ook gebruik van de CEF tabel. Nadeel van CEF is het hoge geheugengebruik: tot enkele honderden bytes per route.

UNIX KERNEL FORWARDING

De manier waarop een Unix kernel te werk gaat bij het forwarden van pakketten houdt het midden tussen fast switching en CEF. Uiteraard gebeurt dit alles in interrupt mode. De datastructuur van de kernel routingtabel is een radix tree: een boom van een vaste diepte waarbij ieder niveau correspondeert met een vast bit in het adres. Hierdoor is de boom altijd gebalanceerd en zijn voor het zoeken alleen logische bit-operaties nodig en geen vergelijkingsinstructies. (De Cisco route caches en CEF tabel hebben vergelijkbare eigenschappen.) Net als bij CEF bevat de kernel routingtabel alle routes, en ook net als bij CEF kosten individuele routes vrij veel geheugen. In oudere versies van FreeBSD, en mogelijk in andere Unix-varianten, staat er een limiet op de hoeveelheid geheugen die de kernel voor de routingtabel wil gebruiken. De standaardlimiet van 20 MB is te krap om de circa 110.000 routes die via BGP doorgegeven worden op te slaan.

De Unix kernel routingtabel doet dus maar weinig onder voor de CEF tabel in het licht van optimale forwarding performance, en het verschil dat er is wordt ruimschoots goedgemaakt door het feit dat een Unix router ofwel een veel snellere CPU heeft dan een Cisco, ofwel veel goedkoper is. (Vaak beide.) In praktijk houdt een niet al te achterhaald Unix systeem makkelijk meerdere fast ethernet interfaces bij, en een snel systeem zelfs een gigabit interface. Dit is vergelijkbaar of zelfs iets beter dan de performance van een Cisco 7200.

Bij gigabit-snelheden beginnen de beperkingen van de PCI-bus merkbaar te worden. In theorie kan een standaard 32-bit PCI bus op 33 MHz 1056 megabit aan. In praktijk lukt het uiteraard niet om 95% van de maximale capaciteit (in één richting) te benutten, nog afgezien van het feit dat veel PCI bussen op 25 MHz draaien. Een 64-bit 66 MHz PCI bus geeft dan meer lucht. Wel moet de gemiddelde pakketgrootte op een redelijk niveau blijven. Met pakketten van 1500 bytes is 1 Gbps ruim 81 duizend pakketten per seconde, maar met de ethernet minimum pakketgrootte van 64 bytes niet minder dan 1,48 miljoen, wat 18 keer zoveel CPU-tijd kost.

Zebra vs Cisco

Cisco is een bedrijf dat nu bijna 20 jaar routers bouwt. Op een enkele uitzondering na draaien deze allemaal Cisco's Internetwork Operating System (IOS). Het IOS image voor een router bestaat uit een enkele file die bij het booten uit flash geheugen geladen wordt. Zodra de router draait is deze via een console of telnet (sommige modellen ook SSH) te managen. IOS kent geen executables of zelfs maar een filesystem in de gebruikelijke betekenis van het woord. Zowel gewone commando's (show, clear, ping, enzovoort) als configuratiecommando's (ip address x.x.x.x, router bgp yyyy) worden direct uitgevoerd. De router is in staat de actieve configuratie weer terug te vertalen naar een lijst configuratie-commando's, die dan als configuratiefile in het NVRAM of op het netwerk opgeslagen kan worden. Cisco routers routeren uiteraard IP, maar ook tal van bekende en minder bekende protocollen zoals IPX, Appletalk, Apollo Domain, DECnet, OSI CLNS en IPv6. De beschikbare IP routingprotocollen zijn RIP (versies 1 en 2), OSPF, Cisco's eigen IGRP en EIGRP en het van OSI afkomstige IS-IS als interne

protocollen, en naast BGP4 ook nog het verouderde BGP3 en EGP als externe protocollen.

Zebra is een verzameling routing-daemons die ieder één protocol implementeren, vergezeld van de "zebra" daemon die zorgt voor het uitwisselen van de routing-informatie tussen de verschillende protocollen en de kernel. Het idee voor Zebra kwam in 1996 van Kunihiro Ishiguro, die het samen met Yoshinari Yoshikawa gestalte gaf. Intussen is Zebra een GNU open source project en zit versie 1.0 eraan te komen. Alle versies tot nu toe zijn prereleases en moeten dus als betaversies gezien worden. Er is ook een commerciële spin-off onder de naam IP Infusion. Zebra ondersteunt RIP versies 1 en 2 en OSPF voor IP, RIPng en OSPFv6 voor IPv6 en BGP4 voor zowel IPv4 als IPv6. De software is getest onder Free-, Net- en OpenBSD, Linux en Solaris.

Zebra slaat een brug tussen twee werelden. Het installeren van de software gaat volgens het geijkte pad binnen de Unix-wereld: .tar.gz files, configure scripts, make commando' sopstartscripts. De configuratiefiles voor de verschillende daemons hebben echter al een verdacht Cisco-achtige syntax. Maar dat is pas het begin. Zodra de verschillende daemons draaien, zijn ze net als een Cisco router via telnet te managen.

De verschillen

Ondanks alle opervlakkige gelijkenissen is Zebra natuurlijk geen IOS-kloon. Zaken als interface-namen blijven de gebruikelijke conventies houden: xl0 of fxp1 en niet ethernet0 of fastethernet2/1. Daarnaast zijn er kleine verschillen in de manier waarop zaken als command-line editing werken. Omdat elk routingprotocol een eigen daemon heeft moeten de commando' svoor ieder protocol via een aparte telnetsessie gegeven worden. Dit is wat onhandig. Er is wel een "vtysh" utility die dit weer samenvoegt.

Veel van de verschillen tussen IOS en Zebra komen voort uit het feit dat Zebra gewoon veel minder features heeft dan IOS. Vaak maakt dit Zebra veel simpeler en logischer. Bijvoorbeeld, deze configuratiecommando' sactiveren het OSPF protocol voor het klasse C net 192.0.2.0:

```
!
router ospf 123
 network 192.0.2.0 0.0.0.255 area 0
!
```

In BGP gaat dit wat anders, de bits in het mask zijn omgekeerd:

```
!
router bgp 65000
 network 192.0.2.0 mask 255.255.255.0
!
```

Het is wel duidelijk dat de mensen die aan verschillende IOS-features werken veel overleg plegen. In Zebra gaat dit allemaal veel makkelijker en logischer:

```
!
router ospf
 network 192.0.2.0/24 area 0
!
router bgp 65000
 network 192.0.2.0/24
!
```

Iets vergelijkbaars geldt voor accesslists. Cisco kent verscheidene varianten: standaard en extended accesslists, die ieder weer varianten met nummers of met namen hebben. Accesslists zijn van oorsprong bedoeld om pakketten mee te filteren. Ze zijn daarom wat tegen-intuitief bij het filteren van routing-updates. De nieuwe prefixlists zijn hier juist wel voor ontworpen. Zebra hoeft geen pakketten te filteren en implementeert dus eigenlijk alleen de prefixlists, maar voor de volledigheid ook in een hybride accesslist-syntax:

```
!
router bgp 65000
 neighbor 192.168.3.169 prefix-list 192.enz/24 out
!
access-list 192.enz/24 permit 192.0.2.0/24
!
```

Een Cisco vindt dit soort creatieve accesslist-namen uit den boze en zal ook nooit een prefixlist voor een accesslist houden of andersom. Voor de functionaliteit maakt het uiteraard geen verschil. En in deze gevallen is er over het algemeen wel een syntax te vinden die zowel Zebra als IOS accepteren.

Dan zijn er nog kleine implementatieverschillen die onder de verkeerde omstandigheden grote gevolgen kunnen hebben. Een goed voorbeeld hiervan is dat bij Zebra het voldoende is om een network statement aan de BGP configuratie toe te voegen. Het betreffende netwerk wordt dan over BGP "geadverteerd". In IOS is dit echter niet voldoende: er moet ook een route zijn die precies gelijk is aan het BGP network statement.

Prijs en betrouwbaarheid

De prijs van een host-based router is eigenlijk altijd lager dan die van een Cisco met een vergelijkbare performance. Daar komt nog bij dat verouderde, maar verder nog prima te gebruiken PC-hardware zo goed als niks kost. Het zijn de extra features die dedicated routers vaak standaard hebben die een host-based router duur maken, als ze al beschikbaar zijn. Bijvoorbeeld, een Cisco 2600 neemt maar een enkele hoogte-eenheid in een 19" rek. Bij een simpele PC kan de 1U-kast net zo duur zijn als alles wat erin moet. Daarnaast kan goedkoop duurkoop blijken te zijn als de eindafrekening van het energiebedrijf komt: een PC gebruikt aanzienlijk meer stroom dan een kleine dedicated router.

Een goede kwaliteit host-based router hoeft qua betrouwbaarheid in principe niet onder te doen voor een dedicated router. In praktijk valt dit toch meestal wel tegen bij het gebruik van relatief goedkope onderdelen. Die gaan niet direct heel vaak kapot, maar ze zijn toch wat minder betrouwbaar dan merk-hardware die bovendien al in een fabriek in

elkaar gezet en getest is. Het echte venijn zit 'm in de harddisks. Met een rotatiesnelheid van 3600, 5400 of zelfs 7200 toeren per minuut hebben die niet het eeuwige leven.

Installatie en beheer

Een werkende router routeert gewoon. Daarin verschillen host-based en dedicated routers erg weinig. Maar de installatie verschilt als dag en nacht. Een dedicated router is bij wijze van spreken al aan het routeren voordat hij de doos uit is. Mensen die gewend zijn met Cisco's te werken denken bij het woord "installatie" eigenlijk alleen aan het plaatsen van 't apparaat in een 19" rek. Hooguit moeten er nog enkele interfacekaarten ingestoken worden en is de router klaar om zijn configuratie te ontvangen.

Bij een host-based router moet vaak de hele machine in elkaar gezet worden. Dan moet het OS geinstalleerd worden, waarbij over het algemeen toch wel minimaal enkele kernel-opties aanpassingen vereisen, en er soms zelfs een volledige nieuwe kernel gecompileerd moet worden, evenals Zebra zelf. Na installatie van de binary's moet er nog voorzien worden in de juiste opstartscripts.

Vreemd genoeg gruwelt de ene groep mensen van het ene, en de andere groep mensen van het andere. Degenen die met routers opgegroeid zijn moeten er niet aan denken zelf hun software te compileren. Wat als de compiler een fout geeft? Wat als er een nieuwe versie op moet, dan kan je alles opnieuw doen! Maar degenen die met Unix systemen opgegroeid zijn slaken een zucht van verlichting als ze de source in hun handen hebben. Als er dan wat misgaat kunnen ze desnoods zelf de fout opzoeken en verbeteren. Bovendien, wat als er van één module een nieuwe versie op moet, dan wil je toch niet het hele systeem vervangen?

In praktijk zijn deze "levensbeschouwelijke" verschillen vaak belangrijker dan de technische voor- en nadelen van beide typen routers.

Een laatste overweging: er is aanzienlijk meer support voor commerciële produkten dan voor open software. Cisco heeft gigabytes aan handleidingen online staan, terwijl de meest actuele documentatie op de Zebra website over een versie van jaren geleden gaat. Aan de andere kant: de source is de beste documentatie.

Conclusie

Host-based routers missen veel van de uitgebreide functionaliteit van dedicated routers en houden nagenoeg altijd en (kleine) achterstand in betrouwbaarheid. Maar binnen die beperkingen kunnen ze uitstekend voldoen en leveren ze aanzienlijk meer performance voor hetzelfde geld. De besparing is wel minder groot dan hij op het eerste gezicht lijkt omdat er meer tijd (en dus geld) gaat zitten in installatie, beheer en het herstellen van storingen.