Korte uitleg: TCP/IP poorten

Deze *Korte uitleg* gaat verder waar *Korte uitleg: IP adressen* ophoudt. Lees die dus eerst.

We zagen hoe TCP/IP pakketjes met gegevens bij een adres kunnen worden afgeleverd. Maar zo'n adres is een computer systeem en in dat systeem kunnen meer processen zijn die tegelijkertijd met een ander systeem communiceren. Voor welk proces is het pakketje nu bedoeld? Er is dus een verdere adressering nodig. Die adressen heten poorten (ports). Het is een getal van 16 bits (0 - 65535, 0 wordt niet gebruikt). Samen met een IP adres geschreven komt het er achter, door een `:` gescheiden: `130.57.66.6:80`. Binnen een systeem zijn de in gebruik zijnde poortnummers op een bepaald moment uniek. En ieder op een bepaald moment in gebruik zijnd poortnummer hoort bij één proces (een proces kan wel meer poorten tegelijk gebruiken). Dat betekent dat, weer op een bepaald moment in de tijd, de adres:poort combinatie in een computer uniek is. Datzelfde geldt voor de computer aan de andere kant. Het gevolg is dat de totale combinatie uniek is op het Internet (op een bepaald moment, over een uur kan alles weer anders zijn). Deze combinatie adres:poort-adres:poort is de identificatie van wat een "established session" (een vastgelegde sessie) wordt genoemd.

Om enige orde te brengen in het poortgebruik zijn de poorten verdeeld in een aantal groepen:

0 - 1023
: "Well-known ports". Deze worden gebruikt door systeem processen die daarmee allerlei server functies aanbieden. De meeste nummers zijn door IANA uitgegeven aan een vaste service/protocol, bijv. 20 voor FTPnetstat, 25 voor SMTP, 80 voor HTTP, enz. Je kunt de lijst vinden in je eigen system in `/etc/services`. In veel operating systemen mogen zij alleen door root processen worden gebruikt.

1024 - 49151
: "Registered ports". Ook deze worden door IANA uitgegeven en vind je in `/etc/services`. Ze kunnen aangevraagd worden door iedere softwarebouwer die een Internet service wil aanbieden. In tegenstelling tot de well-known ports kunnen zij meestal door gewone gebruiker programma's worden gebruikt.

49152 - 65535
: Alle overige poorten worden o.a. gebruikt door client programma's die een sessie willen opbouwen. Na afloop van de sessie komt de poort weer vrij.

Hoe werkt het?

Aan de server kant (het gaat hier overigens niet over de populaire kreet "server", maar over een proces dat een TCP/IP service biedt) registreert een proces zich bij de Kernel voor luisteren op een poort. Bijvoorbeeld, als je Apache hebt draaien, zal dat proces willen luisteren op poort 80. Dan volgt het

grote wachten op een client die een TCP/IP verbinding wil met poort 80 op dit systeem.

Kun je zien welke processen aan het luisteren zijn op jouw systeem?

```
boven:~ # netstat -tlpn
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address     Foreign Address    State     PID/Program name
tcp        0      0 0.0.0.0:2049      0.0.0.0:*          LISTEN    -
tcp        0      0 0.0.0.0:873       0.0.0.0:*          LISTEN    1464/xinetd
tcp        0      0 0.0.0.0:3306      0.0.0.0:*          LISTEN    1822/mysqld
tcp        0      0 0.0.0.0:111       0.0.0.0:*          LISTEN    1/init
tcp        0      0 0.0.0.0:20048     0.0.0.0:*          LISTEN    1572/rpc.mountd
tcp        0      0 0.0.0.0:50387     0.0.0.0:*          LISTEN    1576/rpc.statd
tcp        0      0 0.0.0.0:21        0.0.0.0:*          LISTEN    1464/xinetd
tcp        0      0 0.0.0.0:34487     0.0.0.0:*          LISTEN    -
tcp        0      0 0.0.0.0:631       0.0.0.0:*          LISTEN    1462/cupsd
tcp        0      0 127.0.0.1:25      0.0.0.0:*          LISTEN    1994/master
tcp        0      0 :::2049           :::*               LISTEN    -
tcp        0      0 :::111            :::*               LISTEN    1/init
tcp        0      0 :::80             :::*               LISTEN    1470/httpd2-prefork
tcp        0      0 :::20048          :::*               LISTEN    1572/rpc.mountd
tcp        0      0 :::36630          :::*               LISTEN    1576/rpc.statd
tcp        0      0 :::42807          :::*               LISTEN    -
tcp        0      0 :::631            :::*               LISTEN    1462/cupsd
tcp        0      0 ::1:25            :::*               LISTEN    1994/master
boven:~ #
```

Overigens zien we hier zowel IPv4 als IPv6 adressen.

Als je de `-n` optie weglaat worden de namen uit `/etc/services` gebruikt in plaats van de poortnummers. Soms makkelijker:

```
boven:~ # netstat -tlp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address     Foreign Address    State     PID/Program name
tcp        0      0 *:nfs             *:*                LISTEN    -
tcp        0      0 *:rsync           *:*                LISTEN    1464/xinetd
tcp        0      0 *:mysql           *:*                LISTEN    1822/mysqld
tcp        0      0 *:sunrpc          *:*                LISTEN    1/init
tcp        0      0 *:mountd          *:*                LISTEN    1572/rpc.mountd
tcp        0      0 *:50387           *:*                LISTEN    1576/rpc.statd
tcp        0      0 *:ftp             *:*                LISTEN    1464/xinetd
tcp        0      0 *:34487           *:*                LISTEN    -
tcp        0      0 *:ipp             *:*                LISTEN    1462/cupsd
tcp        0      0 localhost:smtp    *:*                LISTEN    1994/master
tcp        0      0 *:nfs             *:*                LISTEN    -
tcp        0      0 *:sunrpc          *:*                LISTEN    1/init
tcp        0      0 *:www-http        *:*                LISTEN    1470/httpd2-prefork
tcp        0      0 *:mountd          *:*                LISTEN    1572/rpc.mountd
tcp        0      0 *:36630           *:*                LISTEN    1576/rpc.statd
tcp        0      0 *:42807           *:*                LISTEN    -
tcp        0      0 *:ipp             *:*                LISTEN    1462/cupsd
tcp        0      0 localhost:smtp    *:*                LISTEN    1994/master
boven:~ #
```

Het zijn er altijd meer dan je denkt. Servers zoals ipp (Cups) en smtp (E-mail) hebben waarschijnlijk de meesten. Hier vindt je ook www-http (Apache), ftp,

rsync en een heel rijtje in verband met NFS.

Een client process op dit of een ander system kan nu proberen een sessie op te bouwen met een server proces (dus een poort) op dit systeem. Het client proces moet voor die sessie ook een poortnummer krijgen toebedeeld. Dat wordt er één uit de hoge nummers.

Voorbeeld

Een gebruiker op systeem `10.0.0.156` wil met Firefox een web-pagina gebruiken die vanaf systeem `10.0.0.154` moet komen. Het Firefox process is dus een HTTP client (proces op `10.0.0.156`) en zoekt verbinding met de Apache server (proces op systeem `10.0.0.154` luisterend op poort `80`). Het Firefox proces krijgt daarvoor van zijn Kernel poortnummer `65432`. Als alles goed gaat ontstaat er dan een established session tussen `10.0.0.156:65432` en `1010.0.0.0/14.0.0.154:80`. Dat is een unieke identificatie.

Open poort of niet

Een poort waar geluisterd wordt ("openstaand") is natuurlijk niet alleen om te gebruiken, maar het kan ook een gat in je verdediging zijn. De firewall op je systeem kun je gebruiken om poorten toch te blokkeren (maar het lijkt logischer om dan ook het server process niet te draaien), of om een poort voor bepaalde IP adressen wel/niet te blokkeren. Overigens zijn er vaak in de luisterende processen mogelijkheden om alleen van bepaalde IP adressen connecties aan te nemen. Dat levert niet alleen veel mogelijkheden op om te beveiligen, maar ook veel plaatsen waar je moet gaan zoeken als iets niet werkt. Een stappenplan als clients je server niet kunnen bereiken:

1. Controleer of het process (deamon) loopt, bijv:

```
henk@boven:~> ps -e | grep  cups
 1430 ?        00:00:00 cupsd
henk@boven:~>
```

2. Controleer de configuratie van de deamon op blokkeringen. Bijv. uit een Apache configuratie:

```
 Deny from all
 Allow from 10.0.0
 Allow from localhost
```

   laat alleen conputers uit het `10.0.0.0/24` netwerk en `localhost` toe.
3. Controleer met bovengetoond `netstat` commando of er geluisterd wordt op de juiste poort;
4. Controleer de firewall in je systeem;
5. En als je clients van buiten je LAN wilt bedienen, controleer de router op blokkeringen, firewall functies en port forwarding. Voor port forwarding gaan we naar [I]Korte uitleg: NAT en port forwarding[/I].

Proberen met Telnet

Er is nog een handige test om te kijken of je vanaf system x een open poort op systeem y kunt bereiken: test met `telnet`.

In de `man` pagina van `telnet` vind je dat je, na alle opties, de host opgeeft waarheen je `telnet` wilt doen, met eventueel daarachter in een nieuw veld het poortnummer. Dus niet in één veld `130.57.66.6:80`, maar als twee velden. Als je geen poortnummer opgeeft gaat `telnet` naar poort `23`.

```
henk@boven:~> telnet 130.57.66.6 80
Trying 130.57.66.6...
Connected to 130.57.66.6.
Escape character is '^]'.



Connection closed by foreign host.
henk@boven:~>
```

In het geval hierboven kijken we of we verbinding kunnen leggen met poort `80` (http) op `130.57.66.6`. Dat lukt! Welliswaar komen we niet veel verder omdat wij het telnet protocol gebruiken en de andere kant het http protocol verwacht, en na enkele keren Return toets breekt de andere kant de sessie af. Maar we weten toch dat we het systeem kunnen bereiken en dat het systeem op poort `80` luistert.

```
henk@boven:~> telnet 130.57.66.6 81
Trying 130.57.66.6...
^C
henk@boven:~>
```

In dit geval poberen we poort `81` op hetzelfde systeem. We krijgen geen `Connected to ...` en het geheel blijft hangen tot het wordt afgebroken (met Ctrl-C) of tot een time-out.

```
henk@boven:~> telnet 10.0.0.156 80
Trying 10.0.0.156...
telnet: connect to address 10.0.0.156: No route to host
henk@boven:~>
```

In dit geval kan zelfs het systeem `10.0.0.156` niet worden bereikt. Bestaat het wel? Firewall blokkade?