Labo computernetwerken I – sockets

Elke TCP/IP-verbinding wordt gekenmerkt door een 4-tuple <src IP, dest IP, src port, dst port>. Deze 4 getallen kenmerken de unieke verbinding op het Internet. Zowel op de server als op de client wordt een **`socket`** geopend, beide gekenmerkt door deze 4 getallen. Ten minste: na de 3-way handshake tussen beide nodes. Alle IP-pakketten die aankomen op de node, die voldoen aan de 4-tuple, worden door de socket uit hun TCP/IP-headers gehaald en doorgegeven aan de corresponderende applicatie.
In dit lab bestuderen we deze sockets op een Linux systeem.

Listening Sockets – daemon software

1) ss: show sockets

Een TCP-poortnummer op een server is gebonden aan een socket, die steeds bereid is om een TCP SYN pakket te ontvangen. Men heet dit een **`listening socket`**, met **`ss`** kunnen deze met de optie **`-l`** opgevraagd worden. Alle listening (-l) TCP (-t) sockets vraag je op als volgt:

```
comnet1@www:~$ ss -tln
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:21              0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
```

Bovenstaande server heeft dus poorten actief, die erop duiden dat er naast een HTTP en FTP server, ook een SSH daemon actief is zodat remote inloggen mogelijk is. De voorganger van **`ss`** is **`netstat`**.

2) daemon: server software

Een **`daemon`** is een applicatie die opgestart wordt op een computer, los van eender welke actieve gebruiker. De software is permanent actief, en bereidt bepaalde taken uit te voeren. Als die taken ook met het IP-adres gelinkt zijn, wordt een listening socket aangemaakt die een poortnummer bindt aan een bepaalde daemon. Een daemon beheren gebeurt met het commando **`systemctl`**.

Als e.g. de HTTP server nginx geïnstalleerd is, kan je zijn actieve status raadplegen als volgt:

```
root@cnet:~# systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
   Loaded: loaded (/lib/…/nginx.service; enabled; vendor preset: enabled)
   Active: active (running) since Tue 2022-10-25 14:14:15 CEST; 7min ago
     Docs: man:nginx(8)
 Main PID: 5718 (nginx)
    Tasks: 2 (limit: 2346)
   CGroup: /system.slice/nginx.service
           └─5719 nginx: worker process

Oct 25 14:14:15 cnet systemd[1]: Starting A high performance web server and a
reverse proxy server...
Oct 25 14:14:15 cnet systemd[1]: Started A high performance web server and a
reverse proxy server.
```

Een server starten of stoppen kan je eveneens met het **`systemctl`** commando:

```
root@cnet:~# systemctl start nginx
root@cnet:~# systemctl stop nginx
```

Op de derde lijn van de status wordt vermeld dat de service **`enabled`** is. Dit betekent dat de service automatisch start als de computer opstart. Dit is default zo ingesteld door de maker van de software (vender preset), maar kan je ook wijzigen:

```
root@cnet:~# systemctl disable nginx
root@cnet:~# systemctl enable nginx   # Als je het ongedaan wil maken
```

Je zal dit pas na een reboot merken!

De configuratie van een daemon bevindt zich steevast in de map **/etc**. Als dit gewijzigd wordt, moet de daemon service niet per se volledig herstart worden. Je kan een *herladen* van het configuratiebestand doorvoeren. Stel dat we e.g. het IP-adres van een SSH server inperken tot één welbepaald IP-adres, kan de wijziging nadien ingeladen worden met **systemctl reload**:

```
root@cnet:~# grep Listen /etc/ssh/sshd_config
ListenAddress 10.0.2.15
root@cnet:~# systemctl reload ssh
```

Als we nu de listening socket bekijken merken we inderdaad dat het IP-adres bijgewerkt is:

```
root@cnet:~# ss -ltn
State       Recv-Q   Send-Q    Local Address:Port          Peer Address:Port
LISTEN      0        128           10.0.2.15:22               0.0.0.0:*
```

De status bekijken na een **reload** of **restart** is steeds aangewezen! Fouten die optreden bij het uitvoeren van deze reload kunnen wijzen op fouten in de aangepaste files. Fouten worden in de log-file op locatie **/var/log/syslog** weggeschreven; de status geeft de laatste gelogde boodschappen betreffende een daemon weer.

3) Opdracht server ports

1. Bekijk welke poorten er allemaal actief zijn op jouw VM. Stop de DNS daemon **bind9**, en bemerk het verschil. Welke poorten gebruikt de DNS daemon allemaal?
2. Verwijder het programma 'cups-daemon':

   ```
   apt purge cups-daemon
   ```

   Welke poort was er in gebruik door de software (voor printer-services die we niet gebruiken)?
3. Installeer de software **apache2**. Kan je merken welke poort er actief geworden is?
4. Installeer de software **nginx**. Deze daemon wil echter niet opstarten. Leg uit waarom.
5. Stel de poort van **apache2** in op 8080 – dit kan via het bestand /etc/apache2/ports.conf. Herlaad deze daemon. Kan je de wijziging zien in je listening sockets?
   Zou je nu **nginx** kunnen opstarten? Leg uit.

Intermezzo 1: HTTP op de CLI

HTTP(s) is veruit het meest gebruikte protocol op het Internet. Wie op de CLI tools wil gebruiken om html bestanden of objecten te downloaden van een HTTP server, kan gebruik maken van **curl** of wget.

wget laat toe om met HTTP een bestand van een webserver te downloaden, en op te slaan als bestand.

```
student@cnet:~/Downloads$ wget 157.193.215.171/pearson.png
--2022-10-25 23:17:46--  http://157.193.215.171/pearson.png
Connecting to 157.193.215.171:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3267 (3.2K) [image/png]
Saving to: 'pearson.png'
```

curl laat toe om met HTTP een bestand van een webserver te downloaden, en weer te geven op de CLI. Ideaal om te testen of een webserver bereikbaar is, of om de inhoud verder te parsen. E.g.

```
student@cnet:~$ curl www.ugent.be
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>301 Moved Permanently</title>
</head><body>
<h1>Moved Permanently</h1>
<p>The document has moved <a href="https://www.ugent.be/">here</a>.</p>
</body></html>
```

In dit laatste voorbeeld zien we dat HTTP vanzelf wordt omgeleid naar de HTTPs versie van dezelfde server.

1) Opdracht: capture curl?

1. Start wireshark op binnen je VM, met sudo

```
sudo wireshark
```

2. Terwijl je captured op de interface, vraag je met **`curl`** de ugent pagina op.
   Zie je het HTTP/TCP gesprek horend bij deze applicatie?

Active Sockets – client and daemon software

1) Client ports

Als een host geen enkele TCP verbinding opgestart heeft, zien we dat er op het systeem geen enkele TCP socket actief is met behulp van het programma **`ss`** :

```
student@cnet:~$ ss -tn
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
```

Als de client bvb. een eerste SSH verbinding opzet met de server home.test.atlantis.ugent.be , komt deze als eerste lijn in het overzicht terecht:

```
student@cnet:~$ ss -tn
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 10.0.2.15:35586         157.193.215.170:22      ESTABLISHED
```

In dit voorbeeld herkennen we dat op de server poort 22 wordt gebruikt; de client gebruikt zoals verwacht een poortnummer > 1024.

2) Netcat – testen van listening sockets

Het programma **`netcat`** kan gebruikt worden om een TCP-verbinding te openen met een **`listening socket`**. Het biedt gelijkaardige mogelijkheden zoals **`telnet`**, zij het dat de opties uitgebreider zijn. Hieronder vind je een voorbeeld van hoe je ook met **`netcat`** rechtstreeks POP kan communiceren met een mailserver:

```
comnet1@home:~$ netcat mail.test.atlantis.ugent.be 110
+OK Hello there.
USER comnet1
+OK Password required.
PASS XXXXX
+OK logged in.
LIST
+OK POP3 clients that break here, they violate STD53.
.
QUIT
+OK Bye-bye.
```

Het laat toe om snel te testen of een bepaalde poort op een server geactiveerd is of niet: de optie –vz (zie man page) test gewoon of een poort open is of niet, zonder verdere communicatie:

```
comnet1@home:~$ netcat -vz -w 1 www.standaard.be 80
Connection to www.standaard.be 80 port [tcp/http] succeeded!
comnet1@home:~$ netcat -vz -w 1 www.standaard.be 21
netcat: connect to www.standaard.be port 21 (tcp) timed out
```

3) Netcat – opzetten van tests

Maar **netcat** biedt nog meer mogelijkheden: je kan er ook een **listening socket** mee aanmaken:

```
netcat -v -l -p 6789
```

Het IP adres of het localhost adres van de node kan nu op dit poortnummer aangesproken worden. Wie nu vanuit een andere terminal een **netcat** sessie start met dit poortnummer, merkt dat hij inderdaad verbonden is met deze poort – en alle tekst ingegeven aan de client kant, wordt zichtbaar op de server:

```
netcat localhost 6789
```

netcat is dus een tool waarmee je kan testen of je een bepaalde poort kan bereiken, of niet. Wie een server zoekt op internet, om naar te verbinden, kan gebruik maken van http://portquiz.net/ . Deze server heeft voor quasi elke poort een listening socket gemaakt, waarnaar je kan verbinden. Lukt de verbinding niet, dan weet je dat er gegarandeerd ergens een firewall jouw connectie tegenhoudt.

4) Opdracht actieve sockets

1. Maak een SSH verbinding naar home.test.atlantis.ugent.be ; bekijk voordien en nadien de uitkomst met **ss -tn**. Leg uit vanaf wanneer een poortnummer in gebruik is op een client.
2. Maak nadien een tweede SSH verbinding vanaf dezelfde client. Leg uit a.d.h.v. het resultaat van **ss** hoe de pakketten van deze beide verbindingen door de computers uit elkaar kunnen gehouden worden.
3. Bekijk de **listening sockets** op deze server. Leg van 3 poorten uit welke functie ze vervullen op de server. Hint: bekijk de inhoud van het bestand **/etc/services**.

4. Je installeerde reeds de **nginx** webserver. Kan je met **netcat** testen of hij werkt op jouw systeem? Hoe doe je dat? Kan je met **wget** het index.html downloaden?
5. Je installeerde reeds de **apache2** webserver, en werkte het poortnummer bij. Kan je met **netcat** testen of hij werkt op jouw systeem? Hoe doe je dat?
   Kan je met **wget** het index.html downloaden?

6. Stel een poortnummer open op je computer (**netcat listening socket**), zodat je er vanuit een 2e terminal met **netcat** mee kan verbinden. Kan je dit combineren met input/output redirection (zie vorig labo), zodat je het bestand **/etc/services** kan sturen van de ene terminal naar de andere doorheen deze TCP socket? Hint: http://www.microhowto.info/howto/copy_a_file_from_one_machine_to_another_using_netcat

Sockets – programming (in Python3)

1) TCP server

Een server socket hoeft niet steeds door een applicatie opgestart te worden: je kan ook je eigen socket implementeren – bijv. in Python3. Onderstaande code maakt een socket aan op poort 678:

```
from socket import *

s = socket(AF_INET, SOCK_STREAM)
s.bind(('', 678))
s.listen(1)

while True:
    c, addr = s.accept()
    g = ("Hello %s" % addr[0])
    c.send(bytes(g, encoding='utf-8'))
    c.close()
```

2) TCP client

Je kan eveneens een heel primitieve client programmeren in Python3 die verbindt met deze *listening socket* (op het localhost adres):

```
from socket import *

s = socket(AF_INET, SOCK_STREAM)
s.connect(('127.0.0.1', 678))
s.send('Hello World'.encode())
data = s.recv(1024)
s.close()

print('Received', data)
```

Praktisch moet je natuurlijk eerst de servercode opstarten, voor je de client code kan laten verbinden!

3) Opdracht python

Implementeer beide stukken code in twee aparte python scripts. Pas de poort aan naar 6789. Pas eveneens het IP-adres van de client aan naar het (niet-localhost) IP-adres van jouw Linux VM. Start beide scripts op (in de juiste volgorde) vanaf de CLI. Krijg je onderstaande output?

```
student@cnet:~$ python3 tcp-client.py
Received b'Hello 10.0.2.4'
```

Eens je deze interactie kan uitvoeren:

1. Kan je het TCP gesprek capturen met WireShark? Op welke interface werk je?
2. Kan je de server code aanpassen zodat hij het client poortnummer teruggeeft, i.p.v. het IP-adres van de client?

Intermezzo 2: Secure CoPy (SCP)

Tot slot breiden we onze praktische kennis op het Linux systeem nog uit met het SCP commando. SCP laat toe bestanden te kopiëren doorheen een SSH verbinding, waardoor alle informatie die verstuurd wordt geniet van de encryptie die SSH biedt. SCP kan beschouwd worden als een veilige versie van FTP (hoewel de werking anders is dan het FTP protocol, gezien er geen aparte data verbindingen zijn).

1) Aanmaken "tarball"

Creëer volgende file- en directory structuur op jouw Linux machine:

```
gundams
 wingzero
        heero
        yuy
 deathscythe
        duo
        maxwell
 sandrock
        quatre
        raberba
        winner
 heavyarms
        trowa
        barton
 shenlong
        chang
        wufei
```

Gebruik het commando **`tar`** om van deze structuur een archief te maken met de naam **`gundams.tar.bz2`**

Je zorgt daarbij uiteraard ook voor de nodige compressie met **`bzip2`**.

2) SCP kopiëren

- Stuur dit archief aan de hand van **`scp`** vanop de command line naar de server home.test.atlantis.ugent.be
  (hiervoor kan je de account uit de vorige labo's gebruiken).
  Pak het archief daar terug uit.
- Gebruik WinSCP om het archief te kopiëren van deze server naar een Windows PC en pak daar het archief uit (WinRAR ondersteunt **`.tar.bz2`** files).

Socket scanning: nmap

Tools die **`netcat`** als basis gebruiken, maar niet één maar meerdere poorten georganiseerd aftasten heet men een portscanner. De meest gebruikte software op Linux om portscanning uit te voeren is **`nmap`**.

We kunnen bvb. testen of een webserver actief is op een bepaald IP-adres. De status open betekent zoveel als het kunnen opzetten van een three-way handshake met een listening socket:

```
student@cnet:~$ nmap -p 80,443 157.193.43.50
Nmap scan report for portlvs.ugent.be (157.193.43.50)
Host is up (0.013s latency).

PORT    STATE SERVICE
80/tcp  open  http
443/tcp open  https

Nmap done: 1 IP address (1 host up) scanned in 0.09 seconds
```

Als we dit herhalen op een andere poort, e.g. 22, merken we dat we *geen* status **`open`** terugkrijgen.

```
student@cnet:~$ nmap -p 22 157.193.43.50
Nmap scan report for portlvs.ugent.be (157.193.43.50)
Host is up (0.014s latency).

PORT   STATE    SERVICE
22/tcp filtered ssh

Nmap done: 1 IP address (1 host up) scanned in 0.39 seconds
```

Filtered betekent 'ik heb een syn gestuurd, maar nooit een antwoord gekregen'. Met de grootste waarschijnlijkheid heeft een firewall dit syn pakket gedropped. Erg logisch: een webserver moet voor externen niet via SSH toegankelijk zijn!

Nmap start default met het sturen van een ping bericht als probe. Dit wordt door de meeste firewalls geblokkeerd. De optie '**`-Pn`**' schakelt deze probing uit, en gaat direct over naar het opzetten van syn requests voor de gevraagde poorten.

Wie interesse heeft in de vele mogelijkheden van deze 'reconnaisance tool for hackers': https://securitytrails.com/blog/nmap-commands

1) Opdracht nmap

1. Voer een basisscan uit op **`scanme.nmap.org`**. Welke poorten zijn open?
   Welke poorten werden allemaal getest?
2. Voer een scan uit op de server home.test.atlantis.ugent.be.
   Welke poorten zijn er allemaal actief op deze server?
3. Start WireShark op, en start capturing. Scan met nmap je eigen Linux VM op poort 25. Stop het capturen. Welke status krijg je terug? Kan je dit linken aan de TCP-pakketten die je ziet?
4. Herhaal dit experiment, maar test nu poort 25 op de server. home.test.atlantis.ugent.be.
   Welke status krijg je terug? Kan je dit linken aan de TCP-pakketten die je ziet?
5. Na stap 3 en 4 zou je het onderscheid moeten kunnen maken tussen open en filtered als status. Voer een basis portscan uit op www.meemoo.be, en formuleer welk advies je zou kunnen geven aan de beheerder van de firewall van deze server.