Practicum Sniffers

Jay Moors

Pascal de Bruijn

Inhoudsopgave

Plan van Aanpak

Doelstelling

In het kader van de module Software Architectuur in het 5de semester van de opleiding Netwerk Infrastructuur Design dient er een praktijk opdracht uitgevoerd te worden met als onderwerp “sniffers”. Dit om de toekomstige NID-er inzicht te verschaffen over allerhande problemen die zich voor kunnen doen op een ICT-infrastructuur. Denk hierbij onder andere aan performance- en congestie problemen.

Opdracht situatie

Het doel van het practicum is het meten van de impact die een bepaalde applicatie, in dit geval de postcode applicatie, op de netwerk infrastructuur heeft. Hiervoor zullen alle verschillende opties van de postcode applicatie doorlicht worden om zo, door middel van diverse netwerk sniffers, een inzicht te krijgen in de belasting op het netwerk.

Met de postcode applicatie kunnen diverse query's worden uitgevoerd op een database (een postcode tabel) die op een centrale server staat. Deze database beschikt over circa 400.000 records.

Resultaat

Tijdens het practicum zal worden bijgehouden welke resultaten de sniffers opleveren. Na afloop van het practicum dienen we, voor het eind van het blok, een schriftelijk verslag op te leveren met daarin onze bevindingen en conclusies.

Netwerk Sniffers

Voorafgaand aan het practicum is ons verteld dat we voor het beste resultaat en inzicht in de diverse mogelijkheden van sniffers, minimaal 2 sniffers nodig hadden. We hebben toen gekozen voor de Open Source sniffer: Ethereal, en de Closed Source sniffer: Etherpeek, van WildPackets.

Ethereal

Ethereal is een sniffer gebouwd door een groot aantal developers die elk een eigen doel voor ogen hebben, hierdoor ondersteunt Ethereal een groot aantal protocollen (inclusief InterBase/FireBird), welke ontleed kunnen worden.

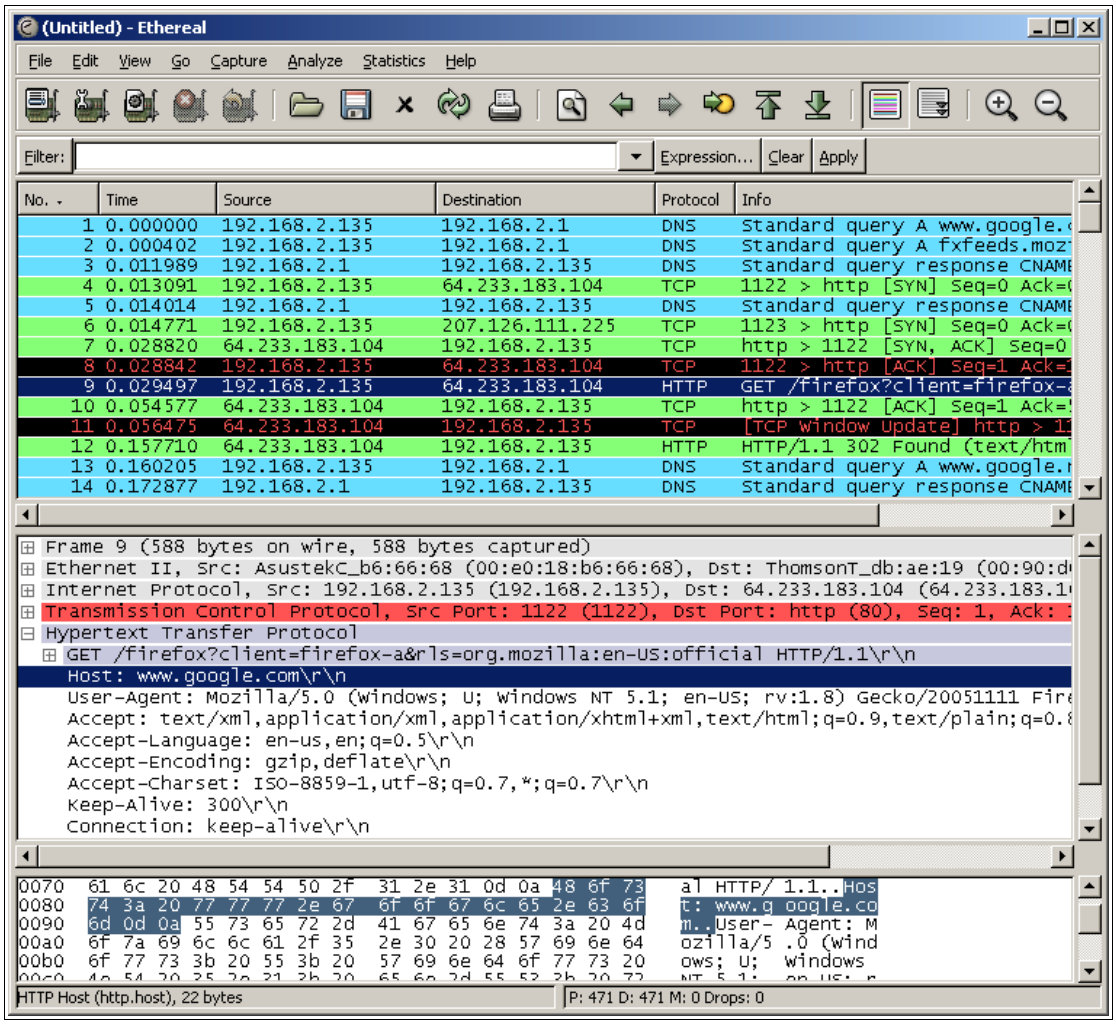
De naam Ethereal is ontleent aan de ontastbaarheid van Ethernet:

> **Ethereal**:
>
> **1 a :** of or relating to the regions beyond the earth
>
> **b :** **CELESTIAL**, **HEAVENLY**
>
> **c :** **UNWORLDLY**, **SPIRITUAL**
>
> **2 a :** lacking material substance **:** **IMMATERIAL**, **INTANGIBLE**
>
> **b :** marked by unusual delicacy or refinement <this smallest, most *ethereal*, and daintiest of birds -- William Beebe

Een ander groot voordeel van Ethereal is dat het platform onafhankelijk is, dus het maakt niet uit welk besturingssysteem je gebruikt, Ethereal werkt even goed op:

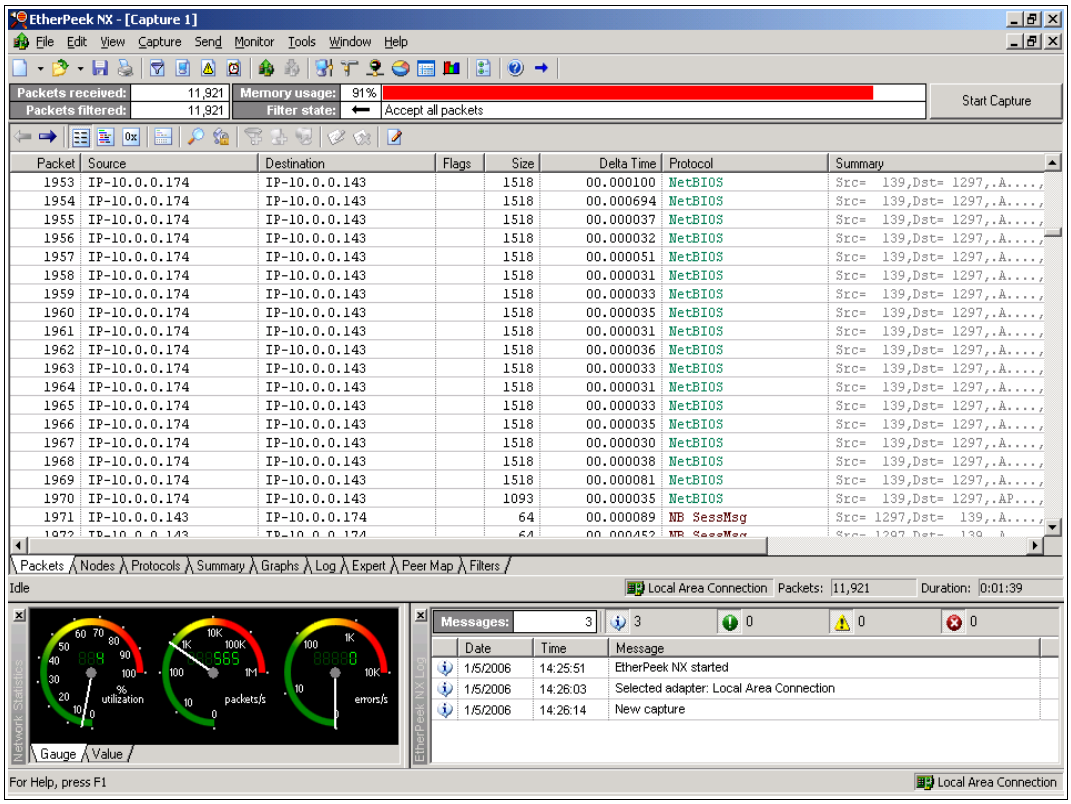
- Microsoft Windows
- Linux
- UNIX: Sun Solaris, IBM AIX, HP-UX, Compaq Tru64, SGI Irix
- BSD
- Apple Mac OS X

Daarnaast biedt Ethereal beide een command line interface en een grafische interface:


(Untitled) - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: Expression... Clear Apply
No. Time Source Destination Protocol Info
1 0.000000 192.168.2.135 192.168.2.1 DNS Standard query A www.google.
2 0.000402 192.168.2.135 192.168.2.1 DNS Standard query A fxfeeds.moz
3 0.011989 192.168.2.1 192.168.2.135 DNS Standard query response CNAME
4 0.013091 192.168.2.135 64.233.183.104 TCP 1122 > http [SYN] Seq=0 Ack=
5 0.014014 192.168.2.1 192.168.2.135 DNS Standard query response CNAME
6 0.014771 192.168.2.135 207.126.111.225 TCP 1123 > http [SYN] Seq=0 Ack=
7 0.028820 64.233.183.104 192.168.2.135 TCP http > 1122 [SYN, ACK] Seq=0
8 0.028842 192.168.2.135 64.233.183.104 TCP 1122 > http [ACK] Seq=1 Ack=
9 0.029497 192.168.2.135 64.233.183.104 HTTP GET /firefox?client=firefox-
10 0.054577 64.233.183.104 192.168.2.135 TCP http > 1122 [ACK] Seq=1 Ack=
11 0.056475 64.233.183.104 192.168.2.135 TCP [TCP Window Update] http > 1
12 0.157710 64.233.183.104 192.168.2.135 HTTP HTTP/1.1 302 Found (text/htm
13 0.160205 192.168.2.135 192.168.2.1 DNS Standard query A www.google.
14 0.172877 192.168.2.1 192.168.2.135 DNS Standard query response CNAME
Frame 9 (588 bytes on wire, 588 bytes captured)
Ethernet II, Src: AsustekC_b6:66:68 (00:e0:18:b6:66:68), Dst: ThomsonT_db:ae:19 (00:90:d
Internet Protocol, Src: 192.168.2.135 (192.168.2.135), Dst: 64.233.183.104 (64.233.183.1
Transmission Control Protocol, Src Port: 1122 (1122), Dst Port: http (80), Seq: 1, Ack:
Hypertext Transfer Protocol
GET /firefox?client=firefox-a&rls=org.mozilla:en-US:official HTTP/1.1\r\n
Host: www.google.com\r\n
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.8) Gecko/20051111 Fire
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8
Accept-Language: en-us,en;q=0.5\r\n
Accept-Encoding: gzip,deflate\r\n
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7\r\n
Keep-Alive: 300\r\n
Connection: keep-alive\r\n
0070 61 6c 20 48 54 54 50 2f 31 2e 31 0d 0a 48 6f 73 al HTTP/ 1.1..Hos
0080 74 3a 20 77 77 77 2e 67 6f 6f 67 6c 65 2e 63 6f t: www.g oogle.co
0090 6d 0d 0a 55 73 65 72 2d 41 67 65 6e 74 3a 20 4d m..User- Agent: M
00a0 6f 7a 69 6c 6c 61 2f 35 2e 30 20 28 57 69 6e 64 ozilla/5 .0 (Wind
00b0 6f 77 73 3b 20 55 3b 20 57 69 6e 64 6f 77 73 20 ows; U; Windows
HTTP Host (http.host), 22 bytes
P: 471 D: 471 M: 0 Drops: 0

Etherpeek NX

Etherpeek NX is een product van WildPackets, zie de screenshot hieronder:



Etherpeek is een professionele netwerk sniffer die van vele handige functies is voorzien. Denk met name aan de duidelijke layout van de gesnifte pakketten en de overzichtelijke grafieken die ermee gemaakt kunnen worden. Tevens zijn de Alarms, Triggers en Notifications erg goed geïmplementeerd. We hebben helaas geen gebruik kunnen maken van de laatste versie van het pakket omdat dit niet voor handen was. We hebben gebruik gemaakt van versie 2.1, terwijl de nieuwste versie 4.0.1 is.

Practicum Opdracht

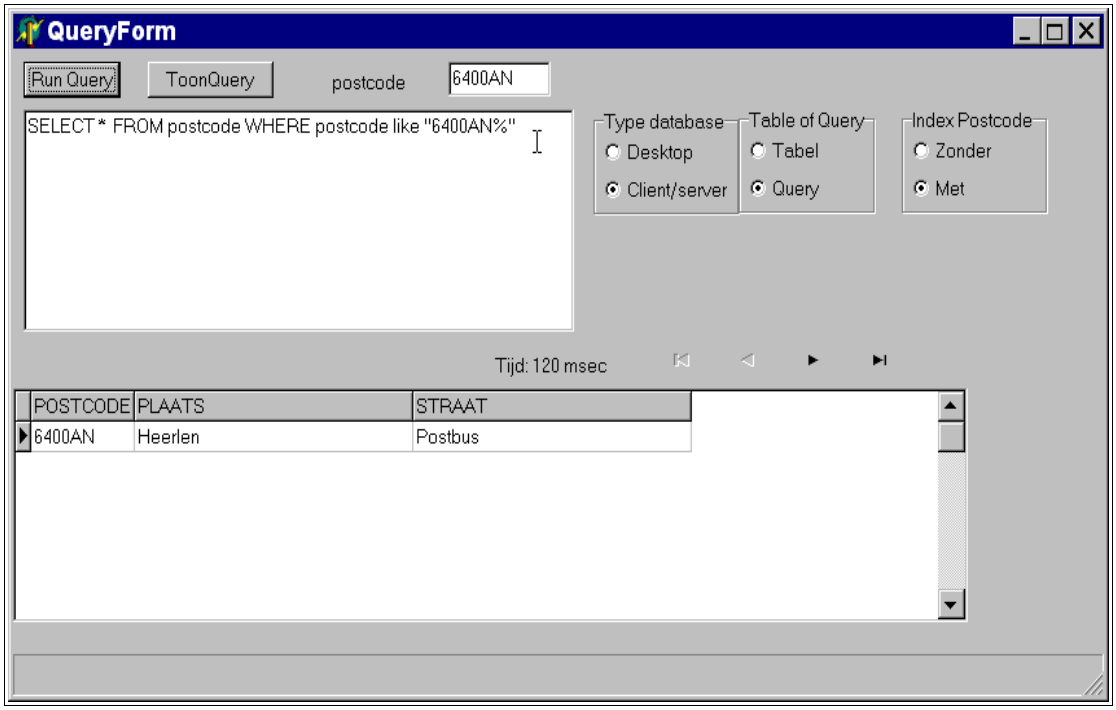
De te raadplegen postcode database bevat ongeveer 400.000 records die bestaan uit de volgende velden:

- Postcode (uniek)
- Plaats
- Straat

De postcode applicatie

Er is een programma in Borland Delphi geschreven voor het raadplegen van de postcode database. In het memo veld kan men bij het uitvoeren van een query een eigen gemaakte query zetten.

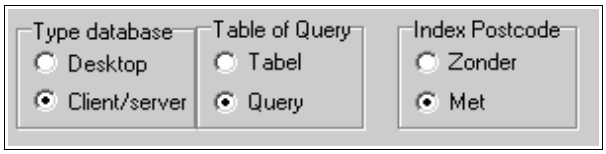
In het edit veld geeft men een geldige postcode in. Klik daarna op de knop “Toon Query” en vervolgens op de knop “Run Query”, zie ook het onderstaand figuur:



We zullen de verschillende opties van dit programma doorlichten en uitleg geven over de door ons vergaarde output. Hierna kunnen we overgaan tot een iets diepere analyse en beoordeling van het programma op de netwerk infrastructuur.

Verschillende opties postcode applicatie

Door middel van de volgende radio buttons kunnen verschillende soorten query's worden uitgevoerd.



Type database: Desktop of Client/server. Als men de Desktop radiobutton kiest, zal gebruik gemaakt worden van de Borland Database Engine, de BDE opent een file via een SMB fileshare, en zoekt via het netwerk voor resultaten. Als men voor de Client/Server radiobutton kiest, wordt een text query naar een echte database server gestuurd (in dit geval FirebirdSQL, een Interbase afgeleide), de database zoekt voor resultaten, en de resultaten worden via het Interbase network protocol naar de client gestuurd.

Table of Query: Query geeft als output een aantal records terug. Table is een alternatieve syntax om query's mee uit te voeren, dit gebruikt doorsnee meer bandbreedte als deze via het netwerk worden uitgevoerd.

Index Postcode: Hierdoor wordt bepaald of bij het doorzoeken van de database gebruik wordt gemaakt van bestaande indexeringen (bijv. B-Tree's, Hashtables, etc.), indexeringen worden gebruikt om het doorzoeken van kritieke data sneller te laten verlopen.

Captures van de query's

We hebben de volgende criteria gebruikt om een overzicht van de resultaten van de query's bij de verschillende opties van het programma te krijgen:

- aantal packets gecaptured

We hebben bewust niet gekeken naar de tijd in milliseconden dat het programma nodig heeft gehad om de query uit te voeren, terwijl dit wel door het programma aangegeven wordt. De reden om hier niet naar te kijken komt door het feit dat we met een hele klas tegelijkertijd verschillende query's uitvoeren. Tevens zullen query's gecached worden, waardoor de tijd voor het uitvoeren van een identieke query aanzienlijk korter wordt. De verkregen tijden zijn daardoor niet relevant.

Table/Query

Als we kiezen voor de optie Client/Server en Query waarbij we een postcode invoeren zien we in de capture dat de query en de respons van de server in clear text over de lijn gaan. Aangezien we gekozen hebben

voor de optie query krijgen we maar enkele resultaten terug.

Desktop/Server	Table/Query	Index	Packets
Server	Table	Ja	247
Server	Table	Nee	235
Server	Query	Ja	26
Server	Query	Nee	28

Als we kiezen voor de optie Client/Server en Table waarbij we een postcode invoeren zien we in de capture dat ook hier de query en de respons van de server in clear text over de lijn gaan. Table werkt met een andere stijl van query's uitvoeren, dit genereert blijkbaar meer dataverkeer.

Indexeringen

Het in- of uitschakelen van indexeringen heeft op een desktop database en een client/server database verschillende effecten.

Desktop/Server	Table/Query	Index	Packets
Desktop	Table	Ja	537
Desktop	Table	Nee	7629
Server	Table	Ja	247
Server	Table	Nee	235

Bij de client/server database heeft dit nagenoeg geen impact: er wordt een query gestuurd naar de database server, de server gebruikt al dan niet de indexeringen om te zoeken, en de resultaten worden terug gestuurd naar client. De gebruiker merkt hier dus niks van (de respons tijd van de server is eventueel wel wat sneller met indexeringen). Let wel op dat de IO wat op de server gegenereerd wordt veel groter is als de indexeringen niet gebruikt worden en de database server serieel door de records heen moet itereren.

Bij de desktop database modus, maakt de applicatie gebruik van de Borland Database Engine, dit is in feite de antieke Paradox database. Paradox was een bestands gerichte database, dus waarbij de applicatie directe toegang had tot het database bestand. Dit kan dus ook via een SMB file share. Net als bij een client/server database maakt het al dan niet gebruik van indexeringen een impact aan de hoeveelheid IO wat op de server gegenereerd wordt. Alleen hier is het concept van 'de server' een beetje vaag. De echte 'server' draait hier namelijk lokaal op de client machine. Als deze machine met indexeringen gaat zoeken worden de

volledige indexeringen (die vrij klein zijn) via SMB over het netwerk verstuurd, en aan de hand van de indexeringen kan bepaald worden precies welke records ook via SMB verstuurd moeten worden. In het geval dat indexeringen niet gebruikt worden, gaat de 'server' dus alle database records doorzoeken, dat betekend dus ook dat alle records via SMB verstuurd moeten worden, en dat dus erg veel netwerk verkeer gegenereerd wordt.

Conclusie

Tijdens het practicum is duidelijk naar voren gekomen dat de Client/Server architectuur significant efficiënter is dan ouderwetse Desktop bestands gebaseerde architectuur.

Ook is tijdens het practicum gebleken hoe voordelig indexeringen kunnen zijn. De indexeringen zijn zowel in de Client/Server architectuur als in de Desktop architectuur voordeliger maar op een andere manier, bij de Client/Server architectuur heeft dit voornamelijk voordelen op de lokale schijf IO van de server, bij de Desktop architectuur heeft dit voornamelijk voordelen op het netwerk verkeer.

Het is lastiger gebleken om duidelijk vast te stellen wat voor verschil Table of Query modus uit maakte, de enige conclusie die we hebben kunnen trekken dat Table modus een andere syntax gebruikt en deze minder efficiënt is.

Aan de hand van dit practicum kunnen wij bevestigen dat de huidige populaire Client/Server architectuur in Query (SQL) modus met Indexeringen het meest efficiënt is, en ten alle tijden aan te raden is.

Bronvermelding

- Practicumhandleiding ARCH in NW5 najaar 2005
- http://www.ethereal.com/
- http://www.wildpackets.com/products/etherpeek/etherpeek_nx/overview