Netwerken Week 1

# 

# **Week 1:**

## Internet

Het internet is een samenstelsel van allemaal Networks

Een Network bestaat uit:

* Toestellen die met elkaar verbinden
* End Systems = Systemen die als host dienen
* Connectivity Devices = Toestellen om andere toestellen met elkaar te laten verbinden
* Applicaties
* Data
* Verbindingen / Links
* Communicatie (via Protocols)

Internet Hierarchy

Diagram

Description automatically generated

## Networks

Network Taken

* Infrastructuur bieden voor applicaties
* Services / Diensten
* Reliable / Unreliable effort
* Monitoring / Rapporting

Protocols

Protocols zijn gemaakte afspraken om communicatie goed te laten verlopen

Includeerd:

* Acties uitvoeren bij start zenden & ontvangen
* Formatting
* Volgorde van zenden & ontvangen
* Eisen van bekabeling
* Acties die ondernomen moeten worden bij ontvangst van Data

Standpunten

Network Edge = Applicaties & Hosts

Access Network = Fysieke Infrastructuur die toegang tot

Backbone / Network Core = Onderling verbonden routers / gateways

Schaalgrootte

|  |  |  |  |
| --- | --- | --- | --- |
| **Naam** | **Schaal** | **Eigendom** | **Beheer** |
| Local Area Network | Locaal | Individueel | Team |
| Metropolitan Area Network | Stad | Organisatie | Team |
| Wide Area Network | Globaal | XXXXXXXXXXXXXX | Meerdere Teams |

Topologie

Graphical user interface, text, application

Description automatically generated

Elk soort Network gebruikt andere protocols & een andere topologie.

Network Edge

Bestaat uit:

* Customer Premises Equipment
* End Systems

Models:

Graphical user interface, application

Description automatically generated

Access Networks

Netwerken die de backbone zijn van een Edge Network

Draadloze netwerken = Access Networks

Access Point / Baste Station nodige component om te kunnen verbinden tussen draadloos en wireless || IEEE 802.11 standaarden

Data Overdracht

|  |  |
| --- | --- |
| Circuit Switching | Packet Switching |
| * Eerst communicatie tussen zender & ontvanger * Verschillende verbindingen op 1 medium * Gereserveerd deel op bandbreedte * Netwerk kan in “Idle” staan * Gegarandeerde performantie | * Gebruik van packets * Volledige bandbreedte gebruikt * Performantie niet gegarandeerd * As-Needed Resource usage * Contention * Congestion mogelijk |

Communicatie Types

Simplex:

* Enkelrichting verkeer
* Partijen hebben een rol
* Ontvanger & Verzender
* Rollen veranderen niet

Half Duplex:

* Tweerichtings verkeer
* Ofwel Zender of Ontvanger
* Rollen kunnen wisselen

Full Duplex:

* Tweerichtings verkeer
* Partijen zijn beide zender als ontvanger

Onderscheiding in:

Unicast: 1 node communiceert met exact 1 andere node

Multicast: 1 node communiceert met een groep nodes in het netwerk

Broadcast: 1 node communiceert met alle andere nodes in het netwerk

# **Week 2:**

Netwerkmodel = visueel beeld van hoe het system opgebouwd is.

Netwerk bestaat uit veel delen:

* Hosts
* Switches
* Routers
* Fysieke verbindingen
* Applicaties
* Protocols
* ...

Netwerkmodel 🡺 Beter overzicht == korter werk == minder problemen

Lagenschema = protocolstack == schema van lagen, elke laag heeft andere regels

2 soorten models:

****ISO/OSI model (theoretisch):

**Applicatielaag**: gedrag van network applicaties

**Presentatielaag**: interpretatie data

**Sessielaag**: opzetten verbindingen, synchroniseren datastroom, controleren data

**Transportlaag**: process data-overdracht in toestel

**Netwerklaag**: adressering van hosts op network + routering van data van bron tot bestemming

**Datalinklaag**: beschrijft de data overdracht tussen LAN elementen

**Fysiekelaag**: beschrijft bits, fysieke signalen & media

TCP/IP model:

**Applicatielaag**: gedrag van network applicaties

**Transportlaag**: process data-overdracht in toestel

**Netwerklaag**: adressering van hosts op network + routering van data van bron tot bestemming

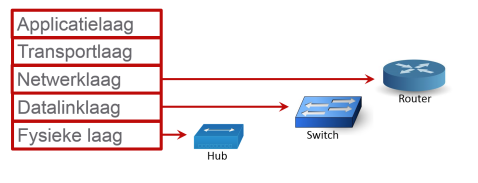
**Datalinklaag**: beschrijft de data overdracht tussen LAN elementen

**Fysiekelaag**: beschrijft bits, fysieke signalen & media

Voorbeelden protocollen per laag

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| APPLICATIE | TRANSPORT | NETWERK | DATALINK | FYSIEK |
| HTTP  FTP  SMTP  IMAP | UDP  TCP | IPv4  IPv6  OSPF  EIGRP | Ethernet  802.3  802. 11a-b-g | ISDN  DSL  100BASE-FX  1000BASE-T |

Netwerkcomponenten



Encapsulatie:

Data sturen van bron naar bestemming effecten:  
 - Data start bij applicatie laag en loopt alles door  
 - Data van beneden naar boven komt toe aan fysieke laag

Elke laag naar beneden geeft extra informatie

Elke laag naar boven verwijdert extra informatie

Toevoegen van informatie == encapsulatie

Data van die je krijgt van een bovenliggende laag == payload

Extra info == overhead 🡪 tijdelijk nut & nodig voor communicatieproces

Diagram

Description automatically generatedCollision Domain:

= Hosts verbonden door hub

--> Als 2 hosts gelijk zenden   
 =   
 Collision (botsing signalen)

Diagram

Description automatically generated Verbinding via switch

--> Zorgt dat de collision domains voorbij de switch   
 starten

Diagram

Description automatically generatedBroadcast domain:

Diagram, venn diagram

Description automatically generatedBroadcast domain = reikwijdte van het bericht

Apparte collision domains

Routers laten geen broadcast berichten door

# **Week 3**

**Applicatielaag HTTP**

HTTP: Hyper Text Transfer Protocol = Protocl waarmee webpagina’s worden overgebracht.  
 Gebruikt in Client – Server architectuur

Table

Description automatically generated

4 versies:

HTTP 1.0 🡪 KOMT NIET MEER VOOR  
 HTTP 1.1  
 HTTP 2.0  
 HTTP 3.0

Communicatie:  
 Client vraagt **HTTP request**  
 Server stuurt gevraagde bestant in **HTTP response**  
 Bestanden kunnen **embedded content** bevatten

Hoe weet TCP waar het pakket naartoe doorgegeven moet worden 🡪 **Poortnummer** in de TCP header

**HTTP uniform resource locators:**

Hoe kan de client aan de server vertellen welk bestand door te sturen?  
 Door in HTTP een **Uniform Resource Locator** te vermelden (URL)

Persistent verbinding:

Diagram

Description automatically generated

Non-Persistent verbinding:

Diagram

Description automatically generated

HTTP

= CLEAR-TEXT PROTOCOL

* Opgebouwd uit ASCII
* Human Readable
* \r\n == Carriage return + Line feed(ENTER)

Voorbeeld HTTP Request

Text

Description automatically generated

HTTP Request Methods:

* GET: Iets uit specifieke URL ophalen
* PUT: Iets op specieke URL wegschrijven
* POST: Iets op specieke URL path aanmaken
* DELETE: Iets op specieke URL vewijderen
* CONNECT: Om client request door proxy of te handelen

Opbouw HTTP Response:

Text

Description automatically generated1ste lijn =   
Statuslijn (statiscode & beschrijving[PHRASE])

Text

Description automatically generated

In RFC kan je de statuscode vinden

Startcijfer geeft weer waar fout ligt

* 1xx: Informationeel
* 2xx: Succesvolle transacties
* 3xx: Transacties waar redirection is
* 4xx: Transactie die foutloopt omwille client error
* 5xx: Transactie die foutloopt omwille server error

HTTPS

Hypertext Transfer Protocol Secure

Uitbreiding op HTTP

Gebruikt om beveiligde verbindingen te maken

!SECURITY!

Protocols: cryptografische protocollen op transportlaag

* + TSL = Transport Layer Security
  + SSL = Secure Sockets Layer (ouder)

Authenticatie = Zekerheid hebben dat de communictiepartners

Encrypteren = Ervoor zorgen dat de clear text data versleuteld en onleesbaar is

SSL