# Linux Theorie

## Sessie 4

# Deel 1

# Terminals

# Terminals

## Wat is een terminal?

In oorsprong dom device

keyboard + monitor

die enkel tekst kan weergeven van het systeem

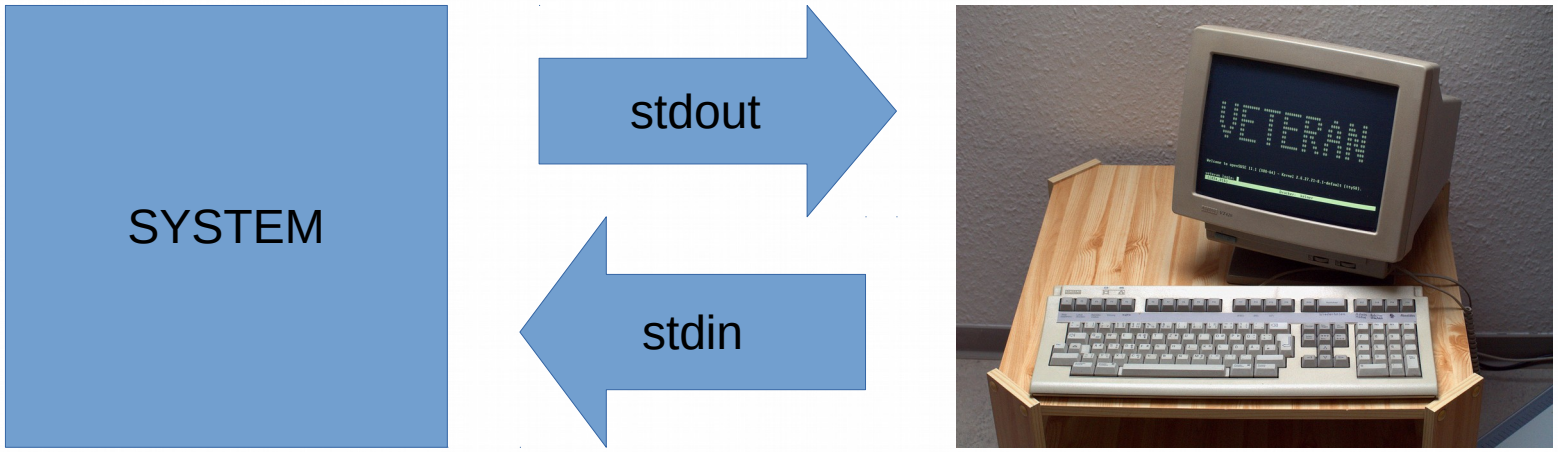
en tekst kan door sturen naar het systeem.

# Terminals


Welcome to openSUSE 11.1 (X86-64) - Kernel 2.6.27.21-0.1-default (tty50).
Druckr: keiner

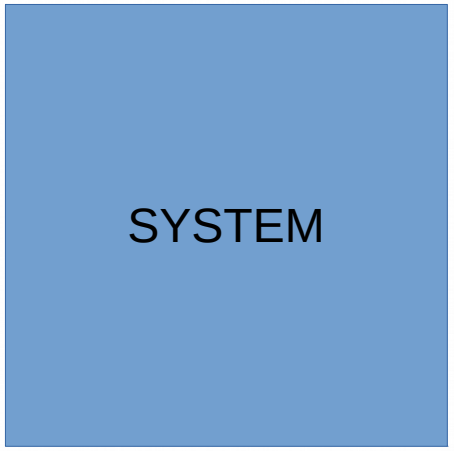
# Terminals

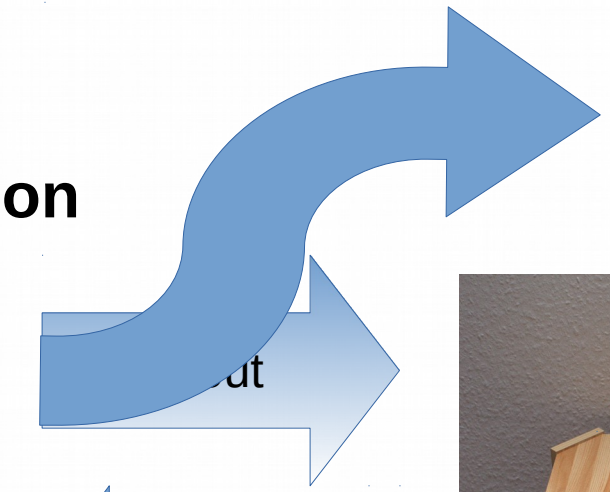
**#include <stdio.h>**

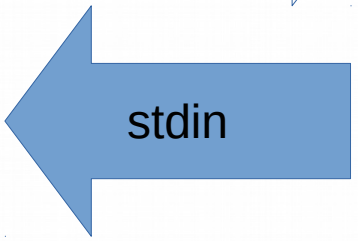


# Terminals

**linux**
**stream redirection**






# Terminals

## Terminal Emulator

- Virtuele terminal binnen de grafische omgeving
- De terminal vensters die we kennen van de labo’s

# Terminals

## Virtuele terminals

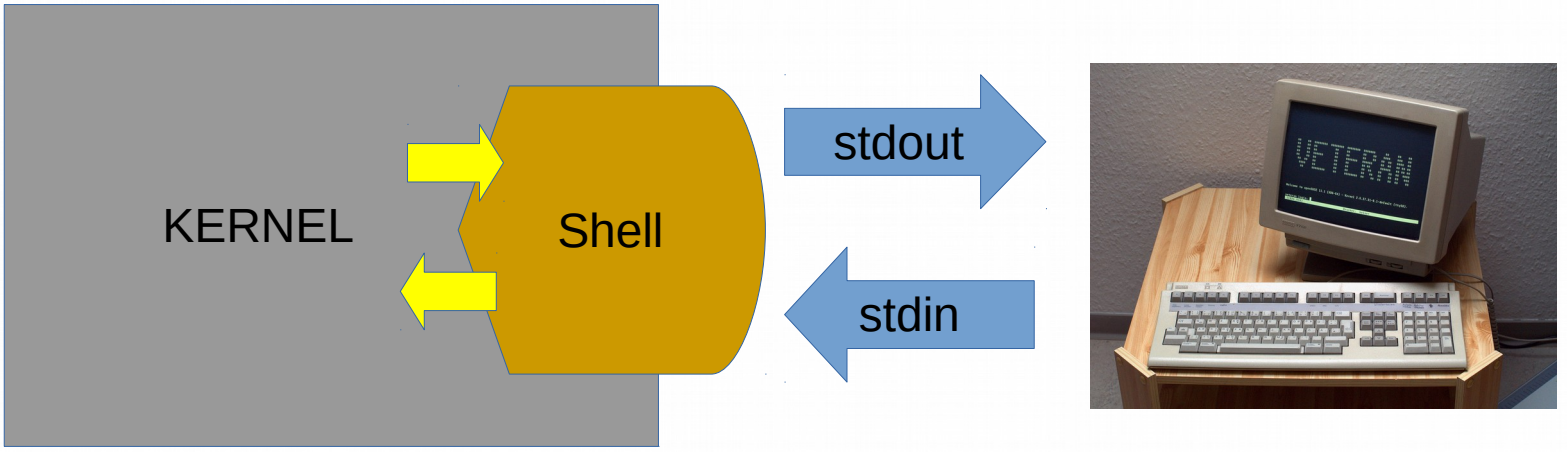
- terminal op bvb PC hardware (zonder GUI)
- bij opstarten in runlevel 3 en hoger
  start Linux meerdere virtuele terminals
- switchen ertussen met Alt-Ctl-F1, F2, …

# Linux Shell

# Shell

## Wat?

shell = commando interpreter tussen de terminal en de kernel



# Shell

## Wat?

- Plaats waar user processen kan starten en stoppen
- Leest en schrijft naar “stdin” en “stdout” ‘bestanden’

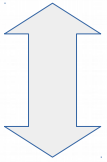
- Shell heeft +- 70 tal ingebouwde commando’s  (Bash)
- Commando **help** geeft lijst + help over ingebouwde commando’s
- Help is op zich reeds een van de ingebouwde commando’s

- Meeste shell commando’s zijn echter externe programma’s
- Ze worden enkel gestart door de shell
- Ze bevinden zich in de /bin directory  (bvb cp, ls, mv ..)

# Shell

## Interpreter ?

Een interpreter vertaalt en
voert compando's één per één uit



een compiler vertaalt een
volledig programma in een keer

# Shell

## Types

- Bourne shell → ‘sh’
- Geschreven door Stephen Bourne voor Unix
- ‘Sh’ was een van de eerste shell’s

- Korn shell → ‘ksh’
- geschreven door David Korn

# Shell

## Types

- Almquist shell → ‘ash’
  → Herschreven Bourne shell met BSD-licence
- Z-shell → ‘zsh’
- C-shell → ‘csh’
- Debian Almquist shell → ‘dash’
  → moderne versie van ash voor debian linux

# Shell

## Types

- ‘Bourne again shell’ → ‘bash’
  → Geschreven voor “The GNU project”
  → **bash** = Default shell in linux

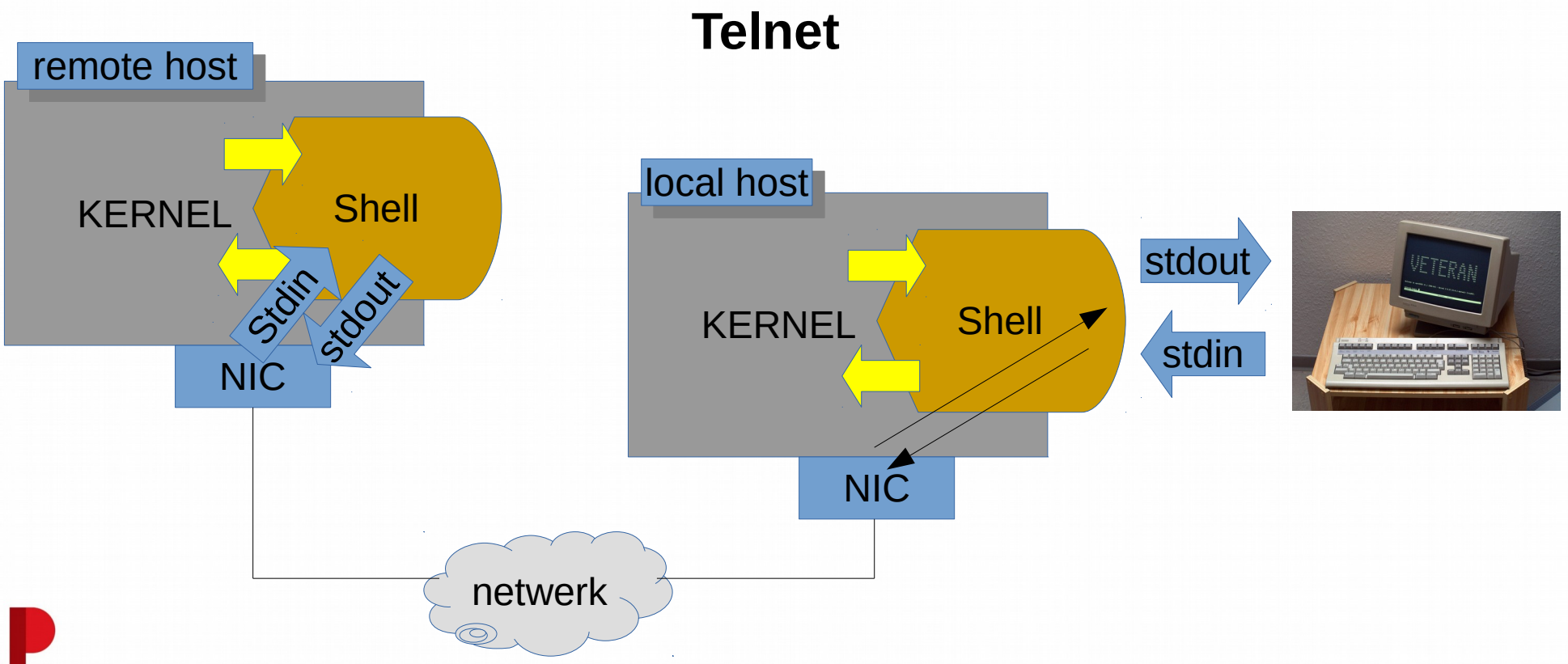
# Shell

**BASH**

éérste programma dat
interface met de werkelijke gebruiker
als doel heeft

# Remote Shell

# Remote Shell



# Remote Shell

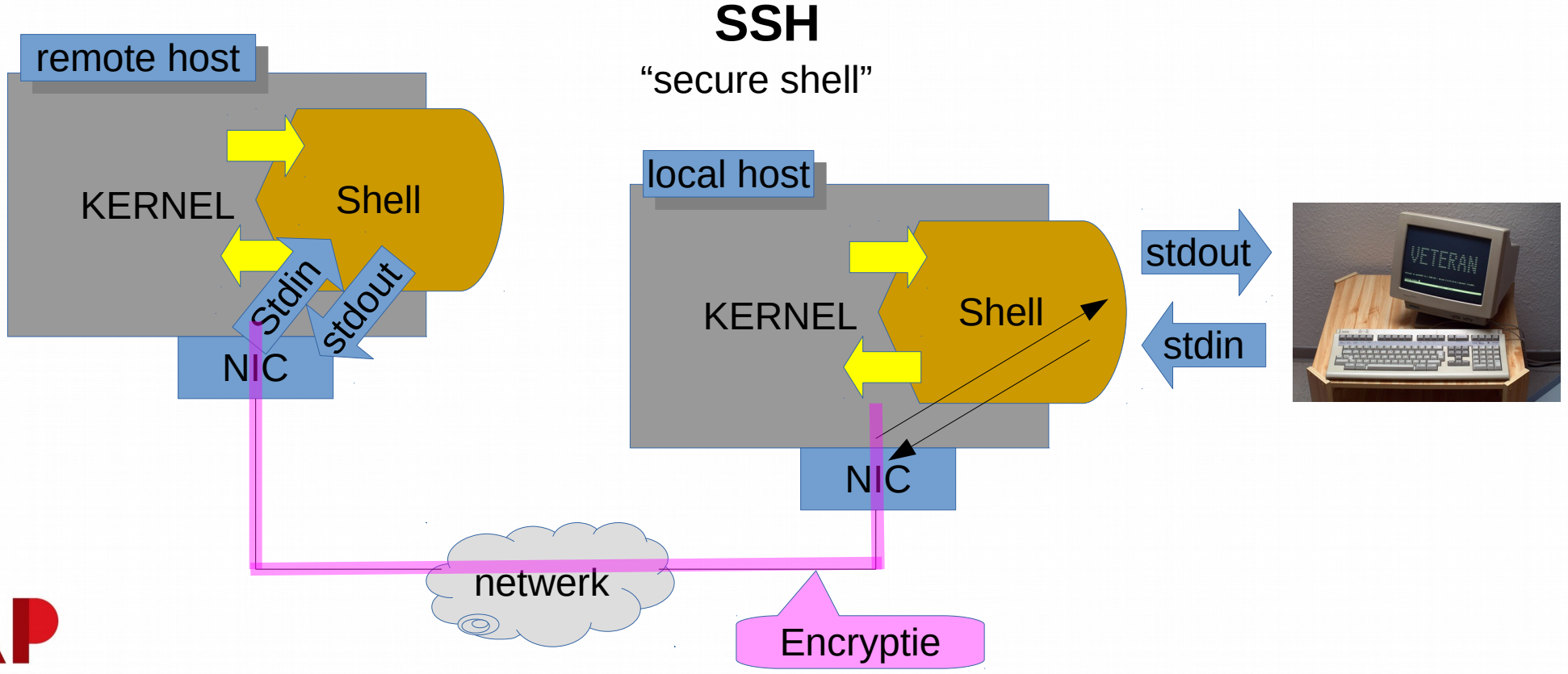
## Telnet

- Telnet-server ↔ Telnet-client

- Oudste remote shell systeem van Linux
- “Plain text” → geen beveiliging

- Wordt nog gebruikt voor directe verbinding met toestellen:
  - via RS232 of RS485 poort op hardware
  - sommige routers & switches

# Remote Shell



# Remote Shell

## SSH

- ssh = “**s**ecure **sh**ell”
- ssh-server ↔ ssh-client

- Linux server implementatie : OpenSSH-server
  (meestal default niet geïnstalleerd)
- Linux client: openssh-client
  (meestal reeds default geïnstalleerd)

- Windows client: - Putty
  - WinSCP

# Remote Shell

## SSH

- Beveiligd door zéér sterke encryptie:
  - Of met paswoord (minder veilig)
  - Of met ‘public-key’ + ‘private-key’ (veiligste methode)

- SSH = meest gebruikte manier voor sys-admin toegang
  tot remote systemen, servers, routers, etc

- Ook voor Raspberry-Pi, Beaglebone, ...

# Encryptie



betekent INDABA

# SSH

## Werkings-principe
## “public-key / private-key”

- Veronderstel computer A en computer B
- We kunnen de data tussen A en B encrypteren
- Als A en B beiden het paswoord kennen kan een verbinding opgezetworden waarbij enkel A en B de data kunnen lezen
- Dit noemen we “symmetrische encryptie”

- Maar hoe kan A dit paswoord aan B laten weten zonder dat iemand anders dit paswood kan te weten komen?

# SSH

## Werkings-principe
## public-key / private-key

- Dit is “asymmetrische encryptie” en vereist een speciaal mechanisme...
- Antwoord op dit vraagstuk kwam er in 1976 door Whitfield Diffie, Amerikaans cryptograaf.
- Maakt gebruik van een slimme wiskundige truuk

# SSH

## Werkings-principe
## public-key / private-key

Video

# SSH

## SSH Commando’s (labo)

- **ssh-keygen**: creëert een set sleutels, public + private key
  *$> ssh-keygen -t rsa*

- **ssh-copy-id**: voegt publieke sleutel toe aan remote machine
  *$> ssh-copy-id -i id_rsa.pub root@192.168.1.1*

- **ssh:** terminal verbinding met de remote computer

- **scp:** “secure copy”
  = cp commando maar tussen lokale en remote machine

# Deel 2

# X-Window System

# X-Window System

## Wat?

- Dé grafische omgeving van Linux en Unix
- Origineel ontworpen in 1984 door MIT
  - Toen gebruikt op mainframes en “mini” computers
  - Versie “X11” kwam op de markt in 1987
  - 30 jaar later nog steeds actueel...
  - Tot en met 2016 → X11 op (bijna) alle Linux computers
- Modulair opgebouwd
- Server-client model

- X11 opvolger is Wayland, reeds 11 jaar in ontwikkeling (2008)
- Sinds nov 2016 in Fedora, Sinds okt 2017 eerste maal in Ubuntu

# X-Window System

## Wat?

- Xorg → dit is de open source implementatie van X11
- Beheer door de x.org foundation ( http://www.x.org)

- Ontwikkeling in samenwerking met freedesktop.org

- Logo:

# X-Window System

## Opbouw:

- X-server, zorgt voor
  - Communicatie met de grafische kaart
  - Keyboard, muis en monitor
  - Creëert een basis grafische omgeving
  - (leeg scherm waar grafische object kunnen op verschijnen)

- X clients
  - zijn applicaties die boven deze basis omgeving draaien
  - geven grafische objecten aan de x-server die ze weergeeft.
  - X-clients kunnen ook remote zijn !

# X-Window System

## Opbouw:

- Window-Manager
  - Zorgt voor de decoratie van de x-client applicaties:
    - balk bovenaan met minimize, maximize, close knoppen
    - kader rond de x-client applicaties
  - Maakt het mogelijk om vensters te verplaatsen op het scherm

- X-fontserver
  - Zorgt voor de verschillende lettertypes in de grafiek

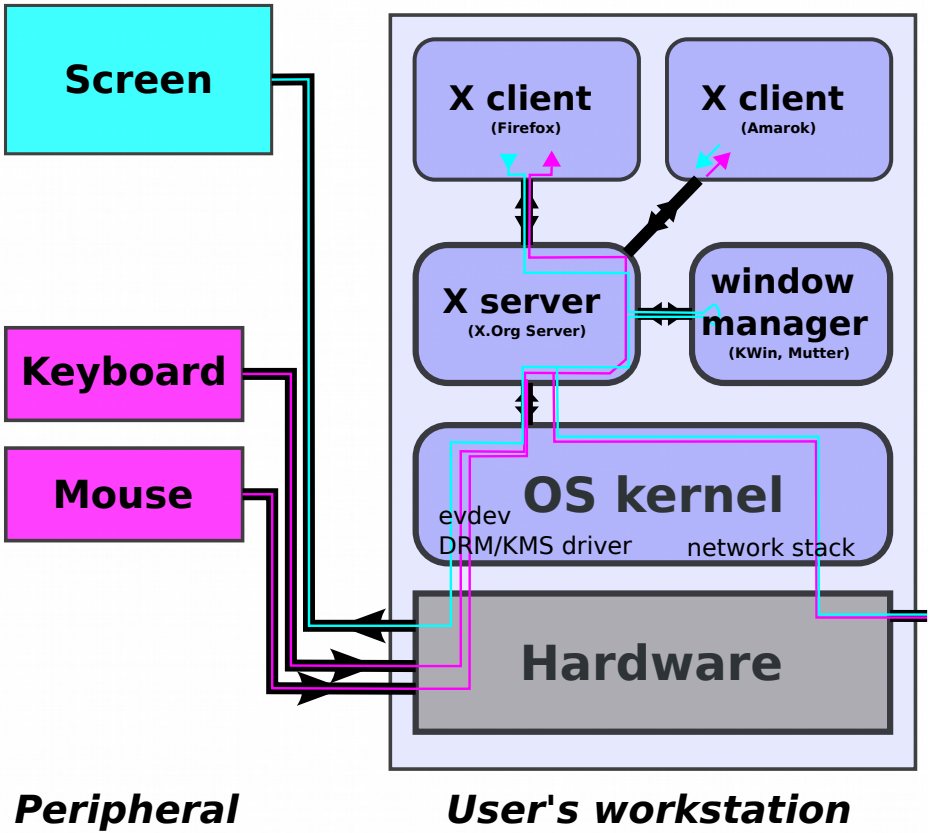
# X-Window System

## Opbouw:

- Populaire window-managers
  - Compiz (ubuntu)
  - Marco (ubuntu-mate)
  - MetaCity
  - Enlightenment
  - Sawfish
  - AfterStep
  - IceWM
  - OpenBox, Fluxbox, Blackbox
  - Xfwm, fvwm, fvwm2, twm, mwm, wm2

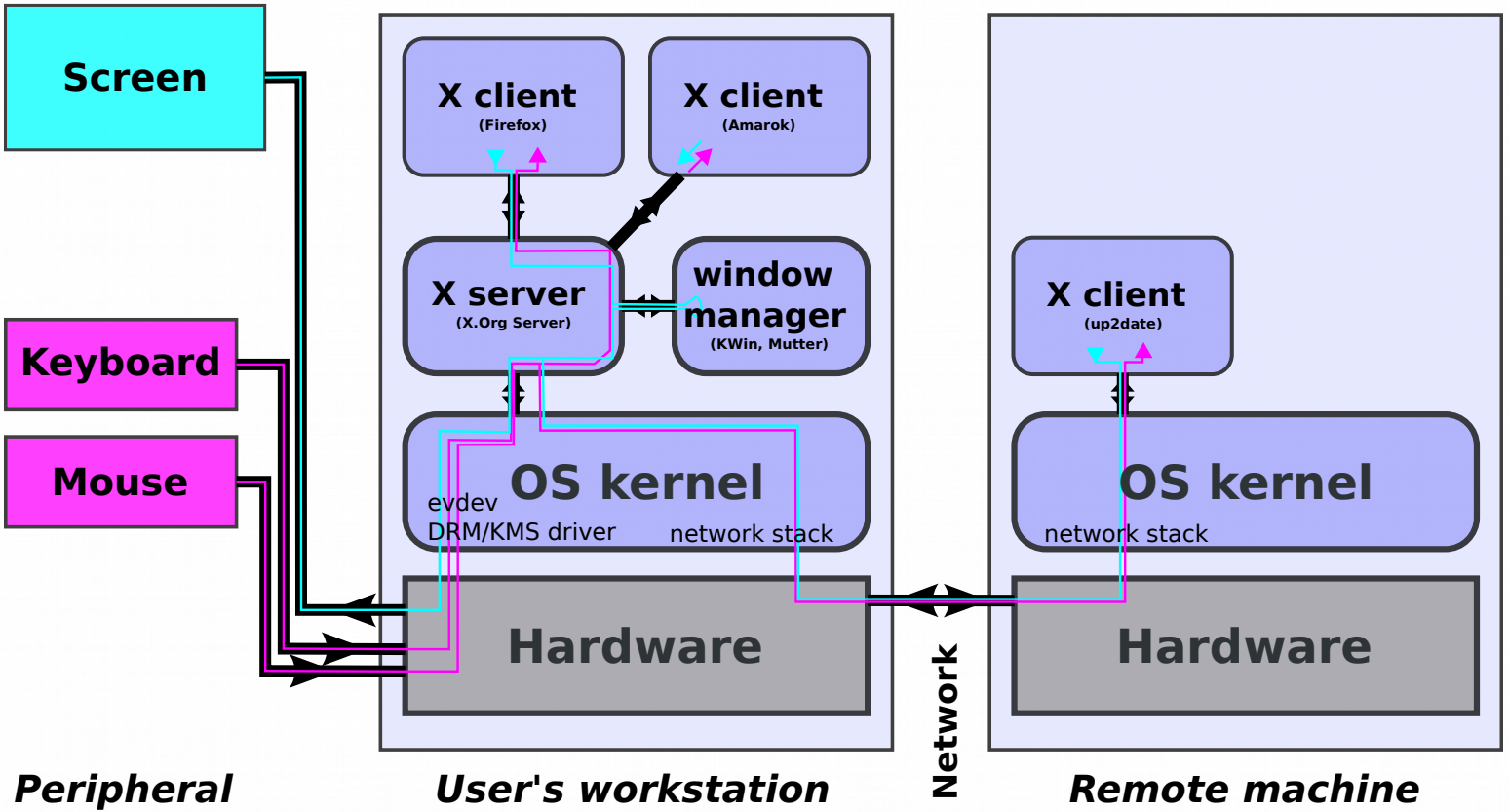
# X-Window System

**The X server manages input and output for even remote clients:**



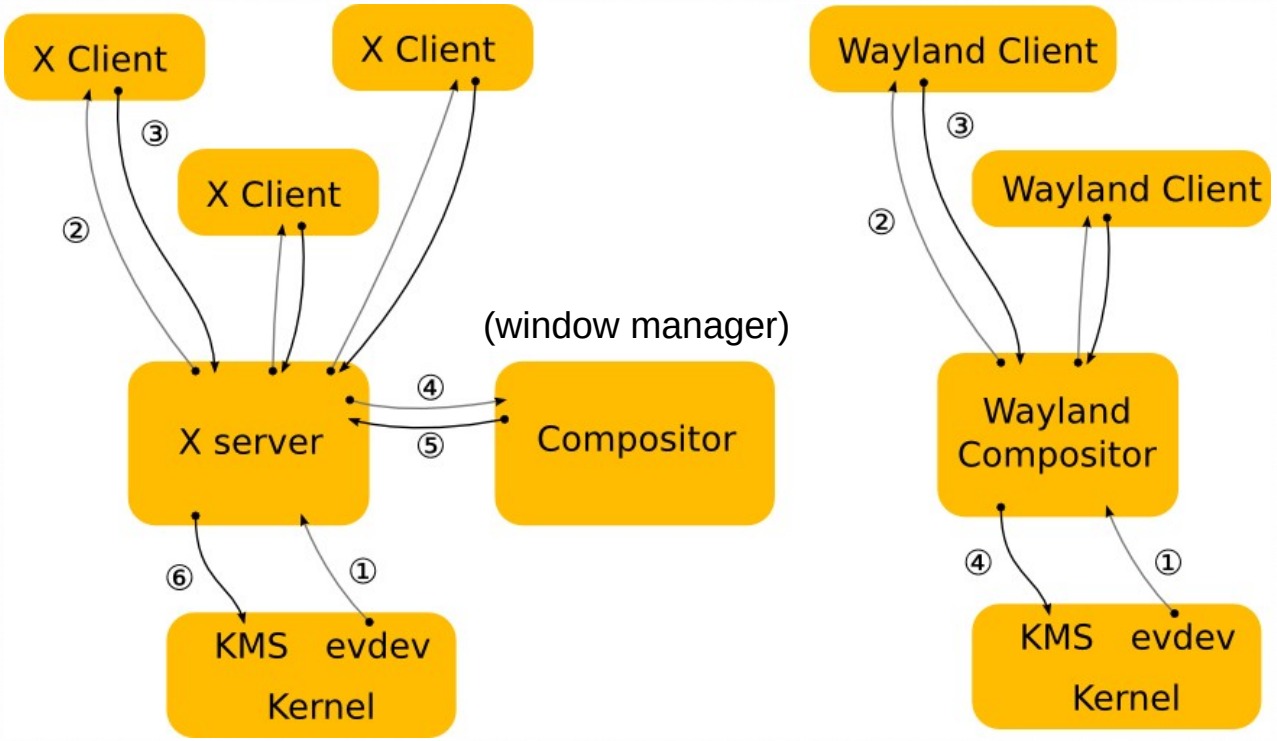
# X-Window System

**The X server manages input and output for even remote clients:**



# X-Window System

## X-window vs Wayland:



# X-Window System

## Praktische werking:

- Bij opstarten in runlevel 5 (graphical.target) zal het init programma automatisch de x-server starten in virtuele terminal 7

- x-server starten vanuit een virtuele terminal:

  *$> startx*

- startx is een front-end van xinit, startx is een script die xinit start, xinit waaruit alle andere x-client processen starten (fork)

- X-server terug afsluiten:

  *$> startx kill*

# X-Window System

## Opbouw:

- Display-Manager
  - Zorgt voor een grafisch login scherm
  - Is eigenlijk reeds een x-client applicatie,
    de éérste x-client applicatie die start na de x-server zelf.
  - De display manager
    - start de window-manager
    - eventueel ook bepaalde x-client applicaties
  - In de display manager kan men bij login kiezen tussen verschillende window-managers
    (of volledige desktop-omgevingen)

# X-Window System

## Opbouw:

- Populaire Display-Managers
  - LightDM (ubuntu)
  - KDM
  - GDM
  - LXDM
  - MDM
  - SliM
  - SDDM

# X-Window System

## Opbouw:

- Typische X-Client applicaties
  - File-manager
    - Konqueror
    - Nautilus ( Ubuntu )
    - Dolphin
    - Krusader
    - Caja ( Ubuntu Mate )
  - Dock
    - Docky
    - Cairo-Dock
    - Plank

# X-Window System

## Opbouw:

- Typische X-Client applicaties
  - Menu
  - Taskbar & Panels
  - Launcher
    - Synapse
    - Launchy,
    - Gnome Do, ...
  - Dashboards
  - Configuration-tools

# X-Window System

## Opbouw:

- Maar ook...
  - Terminal Emulator
    Xterm, Gnome-terminal, Konsole
    Terminator, Guake, Yakuake
  - Tekst editors
    Gedit, Kate, Geany, brackets, eclipse
  - Desktop Calculator
    gcalc, kcalc, gmt, extcalc
  - Desktop Clock
  - ...

# X-Window System

## Desktop Environment

... kortom,

- voor elk onderdeel van de grafische desktop is er in Linux een waslijst van programma's beschikbaar
- Er bestaan daarom vooraf samengestelde collecties van programma's om een volledige functionele desktop te creëren
- Dergelijke collectie noemt men een "**Desktop-environment**"

# X-Window System

## Desktop Environment

- De twee belangrijkste Desktop-Environments:
  - GNOME ( gnome3)
  - KDE ( plasma5)

- Andere heel populaire Desktop-Environments
  - Cinnamon, Unity
  - Mate, Mint
  - XFCE, LXDE
  - Pantheon, Openbox

# Deel 3

# Software & Packages
# Package Managers
# Linux Distributies

# Software & Packages

# Software

## Van bron code tot programma

- Linux is “open source” → bron code
  = leesbare tekst
- Computer begrijpt echter enkel “0” & “1”
- Hoe komen we van tekst tot enen en nullen ?

# Software

## Van bron code tot programma

- Programma = lange reeks combinatie van

*address (plaats) + data (inhoud)*

*1101100010010101..  10101010…*

- Voor de “leesbaarheid” schrijven we de “1” en “0” in hex

*address (plaats) + data (inhoud)*

*0x3FA218F0  0x4B05*

# Software

## Van bron code tot programma

In eerste abstractie zijn deze

*address (plaats) + data (inhoud)*

combinaties opeenvolgende

microprocessor instructies

# Software

## Van bron code tot programma

microprocessor instructies krijgen voor
de mens leesbare namen = opcodes

*Bvb:*

| ***Opcode*** | *Description* | *hex* |
|---|---|---|
| ***LAHF*** | *Load FLAGS into AH register* | *0x9F* |

→ *meestal afkortingen die verband houden met wat ze doen*

# Software

## Van bron code tot programma

elke microprocessor instructie komt overeen
met één of meerdere hex codes (nummering)

*Bvb:*

| *Opcode* | *Description* | ***hex*** |
|---|---|---|
| *LAHF* | *Load FLAGS into AH register* | ***0x9F*** |

# Software

## Van bron code tot programma

een programma geschreven met enkel opeenvolgende opcodes van de microprocessor noemen we “assembly code”

Bvb:

```
mov     edx,len         ;message length
mov     ecx,msg         ;message to write
mov     ebx,1           ;file descriptor (stdout)
mov     eax,4           ;system call number (sys_write)
int     0x80            ;call kernel
```

# Software

## Van bron code tot programma

hogere programmeertalen
proberen dit opnieuw te vereenvoudigen

- Instructies die leesbaar zijn voor de mens
- Deze instructies worden vertaald naar assembly-code door een programma hiervoor geschreven
- Dit noemen we de “compiler”

# Software

**<u>Voorbeeld:</u> “Hello-world” → C Programma**

```
#include<stdio.h>

int main() {
    printf("Hello World");
}
```

# Software

## **Voorbeeld:** “Hello-world” → Assembly Code

```
section   .text
global    _start                ;must be declared for linker (ld)

_start:                         ;tell linker entry point

    mov   edx,len               ;message length
    mov   ecx,msg               ;message to write
    mov   ebx,1                 ;file descriptor (stdout)
```

# Software

## **Voorbeeld:** “Hello-world” → Assembly Code

```
    mov     eax,4                   ;system call number (sys_write)
    int     0x80                    ;call kernel

    mov     eax,1                   ;system call number (sys_exit)
    int     0x80                    ;call kernel

section     .data

msg     db  'Hello, world!',0xa     ;our dear string
len     equ $ - msg                 ;length of our dear string
```

# Software

## **Voorbeeld:** “Hello-world” → Hex Code

*000001b0: 3400 0000 0000 0000 3400 0000 0000 0000*
*000001c0: 0400 0000 0000 0000 51e5 7464 0600 0000*
*000001d0: 0000 0000 0000 0000 0000 0000 0000 0000*
*000001e0: 0000 0000 0000 0000 0000 0000 0000 0000*
*000001f0: 0000 0000 0000 0000 1000 0000 0000 0000*
*00000200: 52e5 7464 0400 0000 100e 0000 0000 0000*
*00000210: 100e 6000 0000 0000 100e 6000 0000 0000*
*00000220: f001 0000 0000 0000 f001 0000 0000 0000*
*00000230: 0100 0000 0000 0000 2f6c 6962 3634 2f6c*
*00000240: 642d 6c69 6e75 782d 7838 362d 3634 2e73*
*00000250: 6f2e 3200 0400 0000 1000 0000 0100 0000*
*00000260: 474e 5500 0000 0000 0200 0000 0600 0000*
*00000270: 2000 0000 0400 0000 1400 0000 0300 0000*

# Software

## **Voorbeeld:** “Hello-world” → Hex Code

*00000280: 474e 5500 b6e1 45c2 4286 939d 7baf 1845*
*00000290: 93be 8aaf f9c5 6a12 0100 0000 0100 0000*
*000002a0: 0100 0000 0000 0000 0000 0000 0000 0000*
*000002b0: 0000 0000 0000 0000 0000 0000 0000 0000*
*000002c0: 0000 0000 0000 0000 0000 0000 0000 0000*
*000002d0: 0b00 0000 1200 0000 0000 0000 0000 0000*
*000002e0: 0000 0000 0000 0000 1200 0000 1200 0000*
*000002f0: 0000 0000 0000 0000 0000 0000 0000 0000*
*00000300: 2400 0000 2000 0000 0000 0000 0000 0000*
*00000310: 0000 0000 0000 0000 006c 6962 632e 736f*
*00000320: 2e36 0070 7269 6e74 6600 5f5f 6c69 6263*
*00000330: 5f73 7461 7274 5f6d 6169 6e00 5f5f 676d*
*00000340: 6f6e 5f73 7461 7274 5f5f 0047 4c49 4243*

# Software

## **Voorbeeld:** “Hello-world” → Hex Code

*00000350: 5f32 2e32 2e35 0000 0000 0200 0200 0000*
*00000360: 0100 0100 0100 0000 1000 0000 0000 0000*
*00000370: 751a 6909 0000 0200 3300 0000 0000 0000*
*00000380: f80f 6000 0000 0000 0600 0000 0300 0000*
*00000390: 0000 0000 0000 0000 1810 6000 0000 0000*
*000003a0: 0700 0000 0100 0000 0000 0000 0000 0000*
*000003b0: 2010 6000 0000 0000 0700 0000 0200 0000*
*000003c0: 0000 0000 0000 0000 4883 ec08 488b 0525*
*000003d0: 0c20 0048 85c0 7405 e843 0000 0048 83c4*
*000003e0: 08c3 0000 0000 0000 0000 0000 0000 0000*

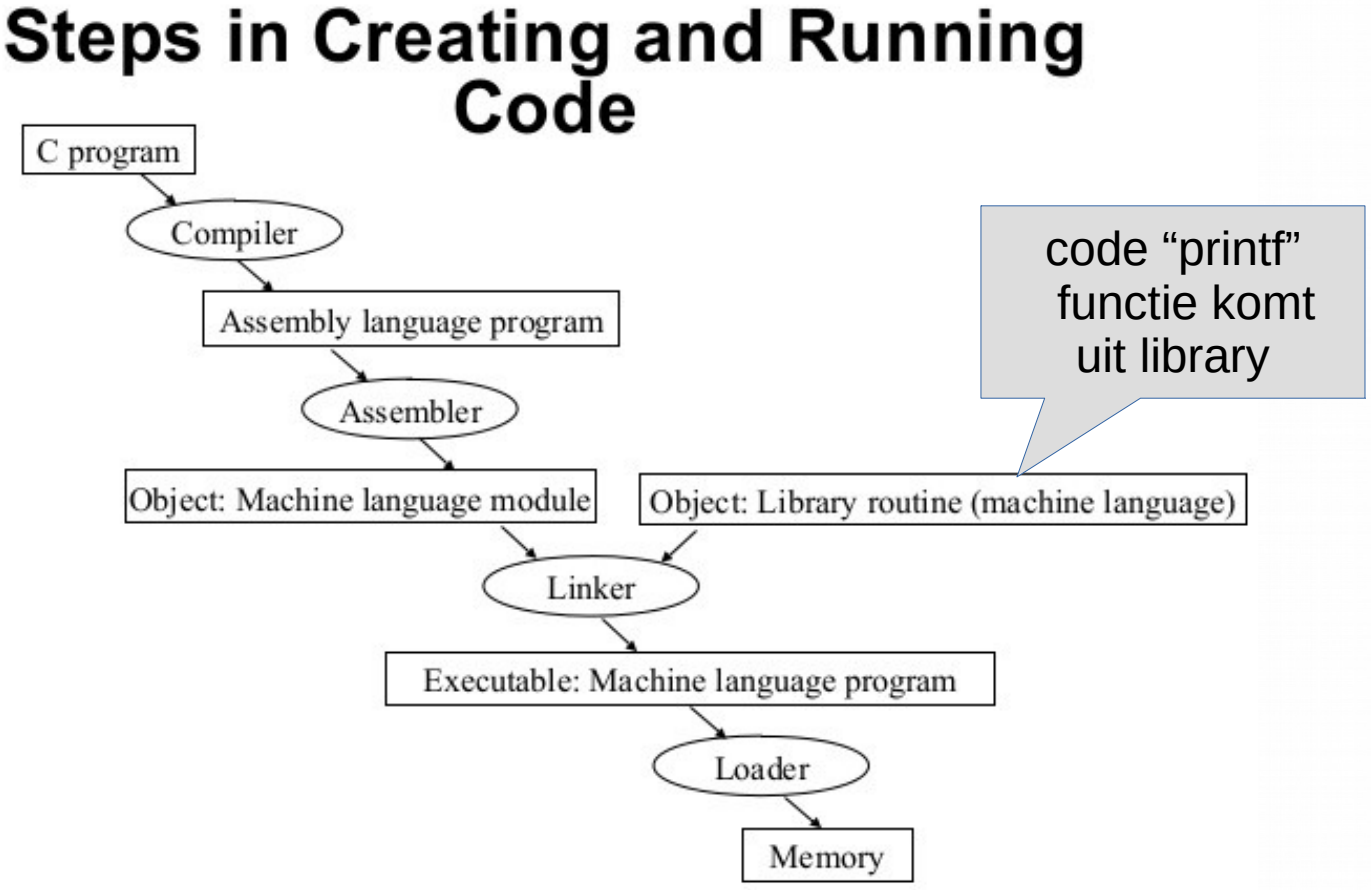
*enz … enz . Nog 20 slides ...*

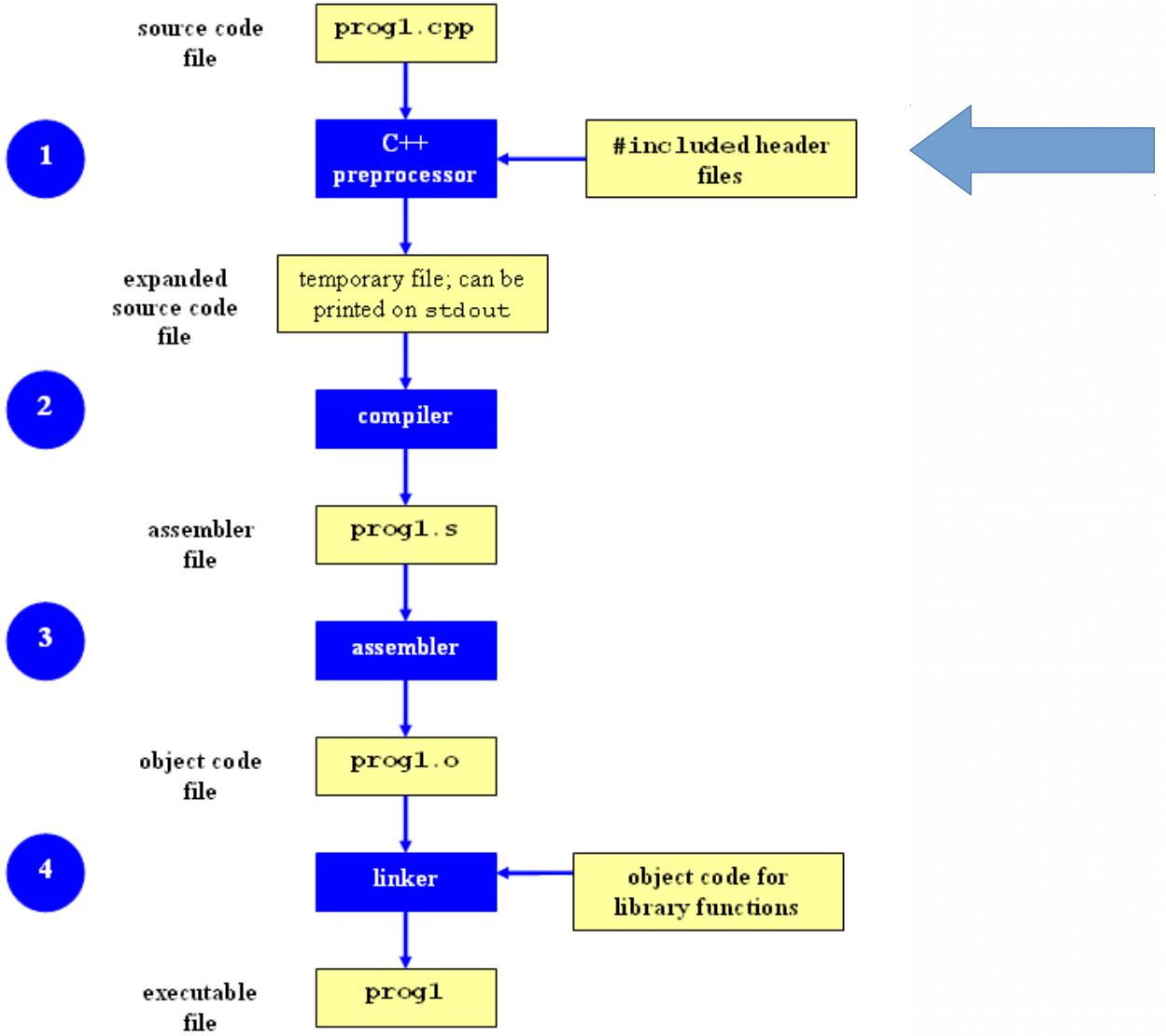
# Software

**Waarom is hex code zoveel langer… ?**

→ *dit is het werk van de “linker”*

# Software


Steps in Creating and Running Code
C program
Compiler
Assembly language program
Assembler
Object: Machine language module
Object: Library routine (machine language)
Linker
Executable: Machine language program
Loader
Memory
code “printf” functie komt uit library


source code file
prog1.cpp
1
C++ preprocessor
#included header files
expanded source code file
temporary file; can be printed on stdout
2
compiler
assembler file
prog1.s
3
assembler
object code file
prog1.o
4
linker
object code for library functions
executable file
prog1

# Software

**Waarom dit hele process ?**

- Verschillen in hardware

# Software

**ARM Assembly**

```
mov     %r0, $1
ldr     %r1, =msg
ldr     %r2, =len
mov     %r7, $4
swi     $0
```

**Intel x86 Assembly**

```
mov     edx,len
mov     ecx,msg
mov     ebx,1
mov     eax,4
int     0x80
```

# Software

**Waarom dit hele process ?**

- Verschillen in hardware
- Efficiëntie → library functies

# Building Software

## Make files & Make

- Software bestaat meestal uit
  - meerdere programma's  (vb bash)
  - Libraries
  - Configuratie bestanden
  - Documentatie
  - …
- Compileren en installeren hierdoor arbeidsintensief

# Building Software

## Make files & Make

- Oplossing is **Make** commando
- Make gebruikt hiervoor een make-file
- Soort ‘template’ bestand die gaat kijken welk systeem, welke directories etc..
- Make ‘maakt’ het programma voor jouw systeem
- wordt vaak vooraf gegaan door “**Configure”**

# Building Software

## Make files & Make

Typische procedure:

- Downloaden bron code
- Vervolgens cd in de bron directory

***./configure***
***make***
***sudo make install***

# Building Software

## Knelpunten

- Compileren kan lang duren
- Vaak bevat een software pakket heel wat software componenten (Bvb Libreoffice)
- volledige linux installatie bestaat uit honderden software pakketten

# Building Software

## Knelpunten

- Installatie van linux vanaf broncode

  → dagen en dagen werk …

  → énkel compilatie tijd reeds meerdere dagen zelfs op moderne multicore PC…

# Building Software

## Oplossing

- Slimme vooraf gecompileerde “packages”
- Package installeer & beheer programma

  = “package manager”
- Verzameling van dit soort packages samen met de nodige automatisch installatie tools

  = “Linux distributie”

# Building Software

## Oplossing

- Enkele bekende linux distributies:
  - Debian, Redhat, Ubuntu, Linux mint
  - Fedora, OpenSuse, CentOS
  - Slackware, Gentoo, Arch, enz…
- Bevatten reeds voorgecompileerd:
  - Linux kernel, ...
  - Diverse terminal applicaties
  - Openssh-server, X-server, Cups (printer-server)
  - Desktop Environment met een plethora aan software zoals LibreOffice, Firefox, etc

# Linux Distributies

## Oplossing

- Slimme pre-gecompileerde packages
  - Red-Hat: **.rpm** → gebruikt door:

    Fedora, OpenSUSE,
    Red-Hat, centOS

  - Debian: **.deb** → gebruikt door:

    Ubuntu, Debian, Linux Mint,
    Devuan, Trisquel, Knoppix, ...

# Linux Distributies

**Installatie van individuele packages:**

- Debian .deb (bvb ubuntu)
  = dpkg commando

  *$> dpkg -i mijnprogramma.deb*

- Red-Hat .rpm (bvb fedora)
  = rpm commando

  *$> rpm -ivh mijnprogramma.rpm*

# Linux Distributies

## nadelen van dpgk & rpm

- Packages zijn meestal op hun beurt afhankelijk van andere packages

  *bvb ‘gedit’ heeft enkele ‘gnome’ packages nodig om te kunnen werken*

- dpkg en rpm houden niet bij wat waar van afhankelijk is en of bvb het verwijderen van een package geen andere programma’s gaat breken.

# Linux Distributies

## Oplossing

- Package installeer & beheer programma
  → “package manager”

  - Red-Hat: **yum**
  - Debian: **apt**
  - Opensuse: **zypper**
  - Arch: **pacman**

# Linux Distributies

## Installatie programma's

- Red-Hat: **yum**

  *$> sudo yum install [programma]*

- Debian: **apt**

  *$> sudo apt-get install [programma]*

- Opensuse: **zypper**

  *$> sudo zypper install [programma]*

# Linux Distributies

## Controleren op update's

- Red-Hat: **yum**

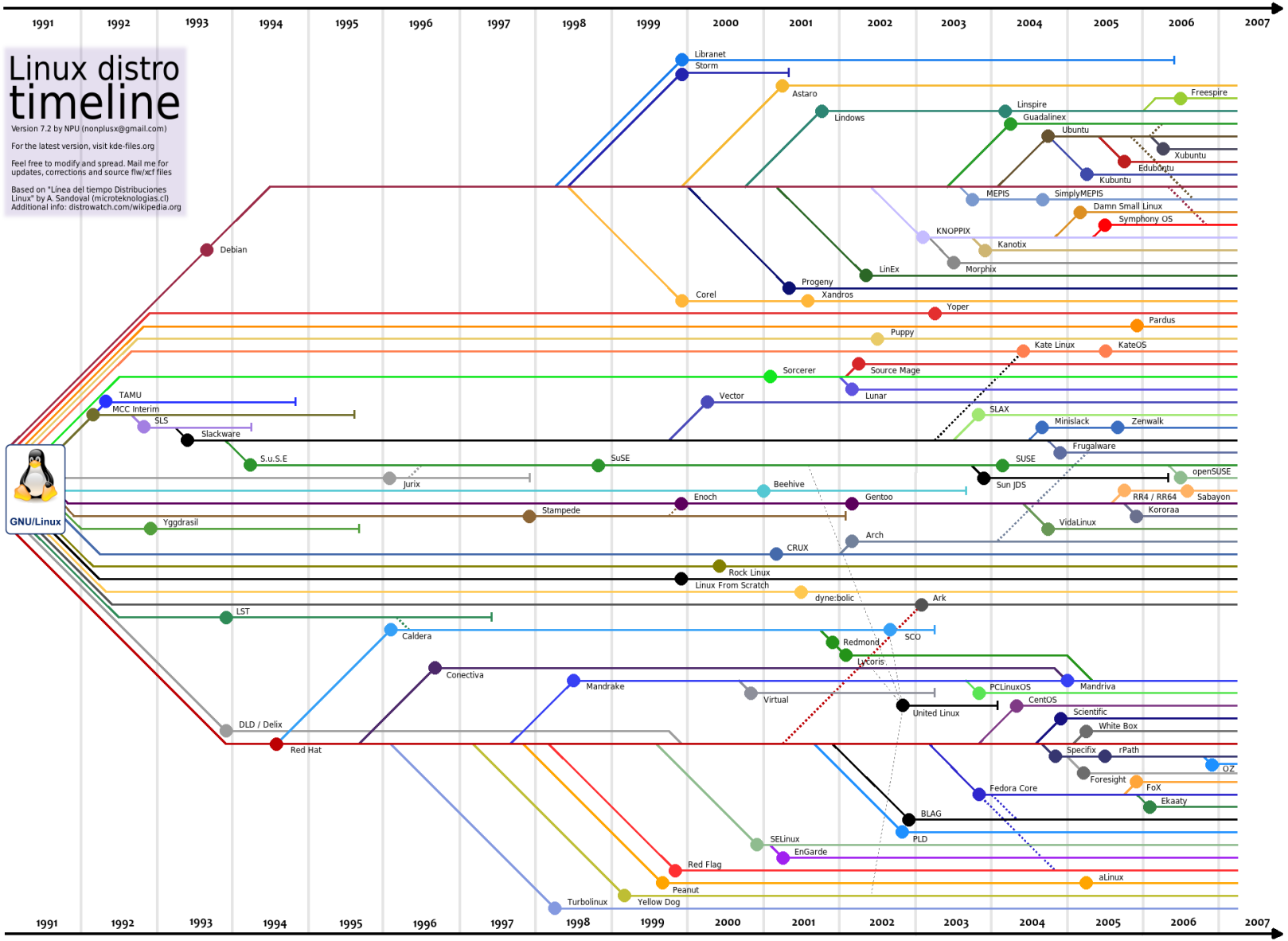
  *$> sudo yum check update*

- Debian: **apt**

  *$> sudo apt-get update*

- Opensuse: **zypper**

  *$> sudo zypper up*



# Linux Distributies

- “Distro’s” zijn vaak een vertakking van andere distributies
- Slackware, Debian, RedHat en Suse behoren tot de oudste distributies
- Sommigen komen en gaan...
- OpenSource zorgt voor “recyclage”: goede ideën komen vaak terug in een andere of nieuwe distro terecht

# Linux Distributies

**Demo**
**Distrowatch.org**