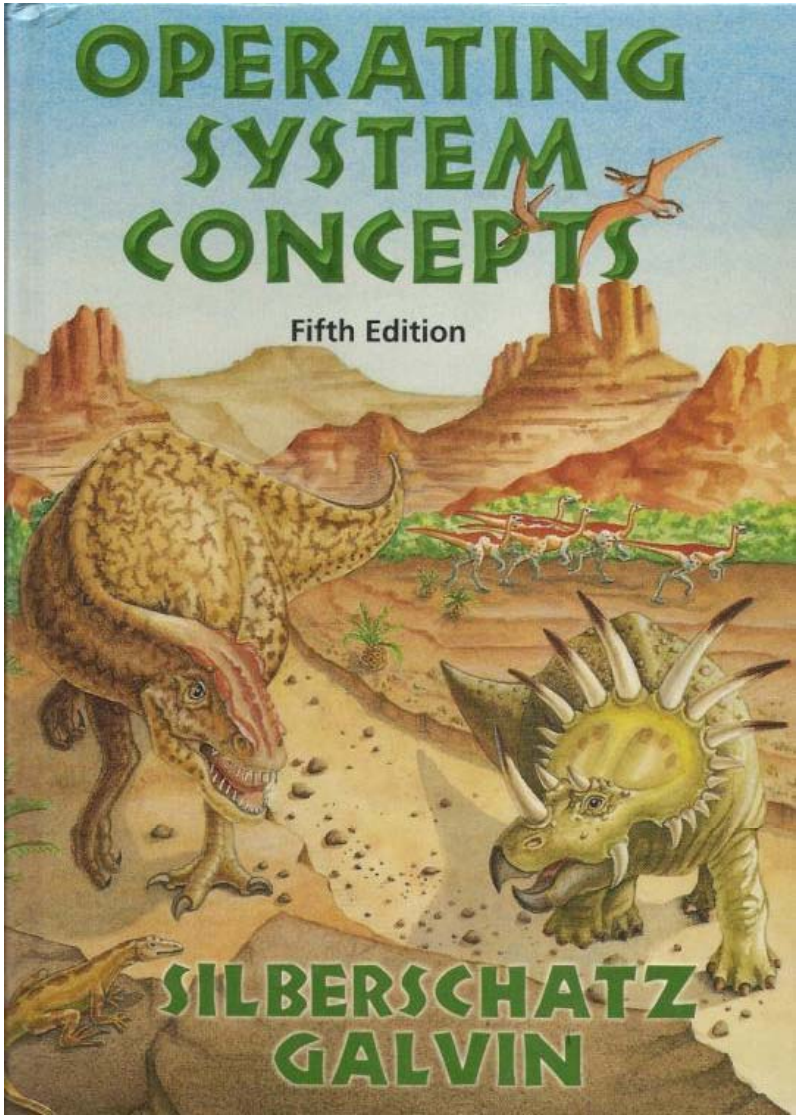
De evolutie van operating system beveiliging

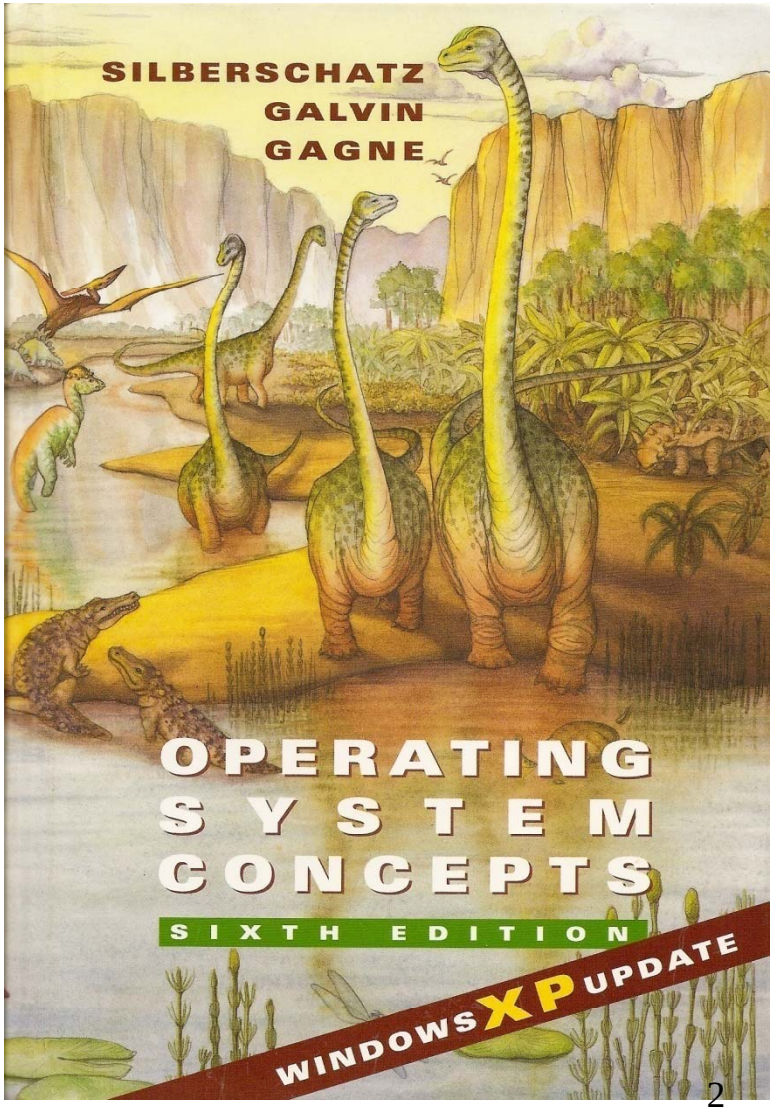
Erik Poll

Digital Security group
Radboud Universiteit Nijmegen

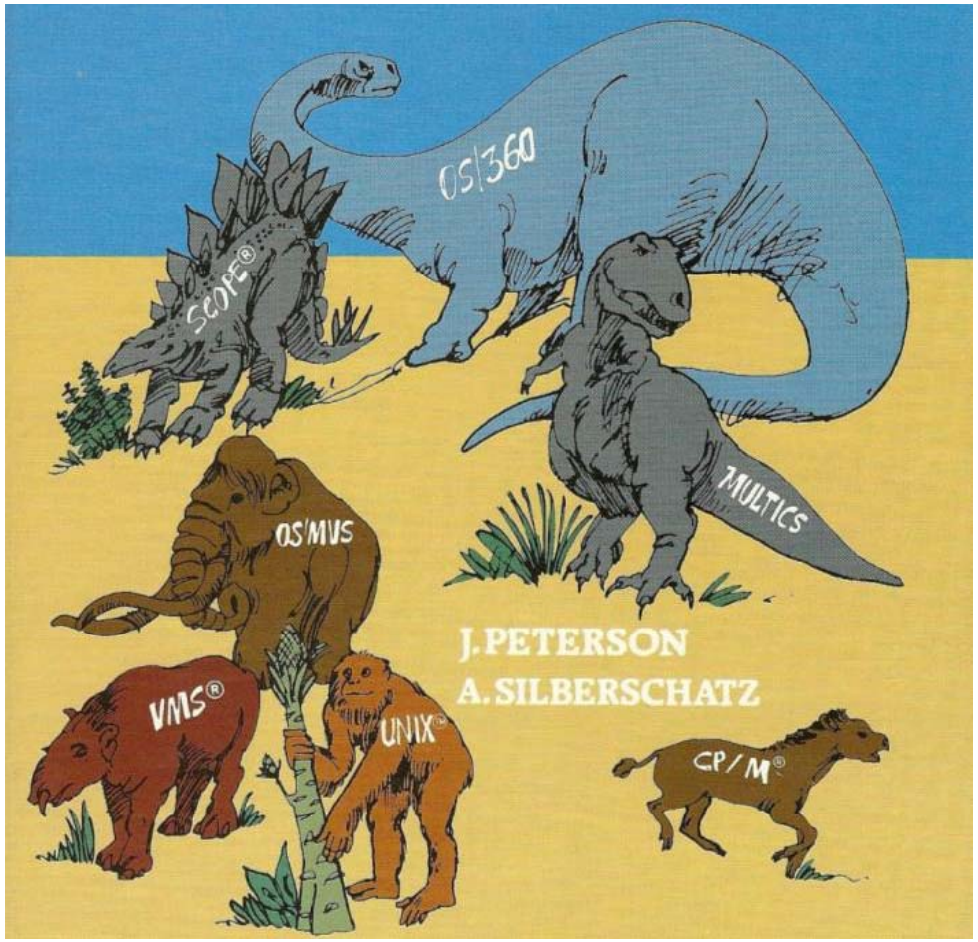
k3rckhoffs
1nst1tute

Moderne Operating Systems ?





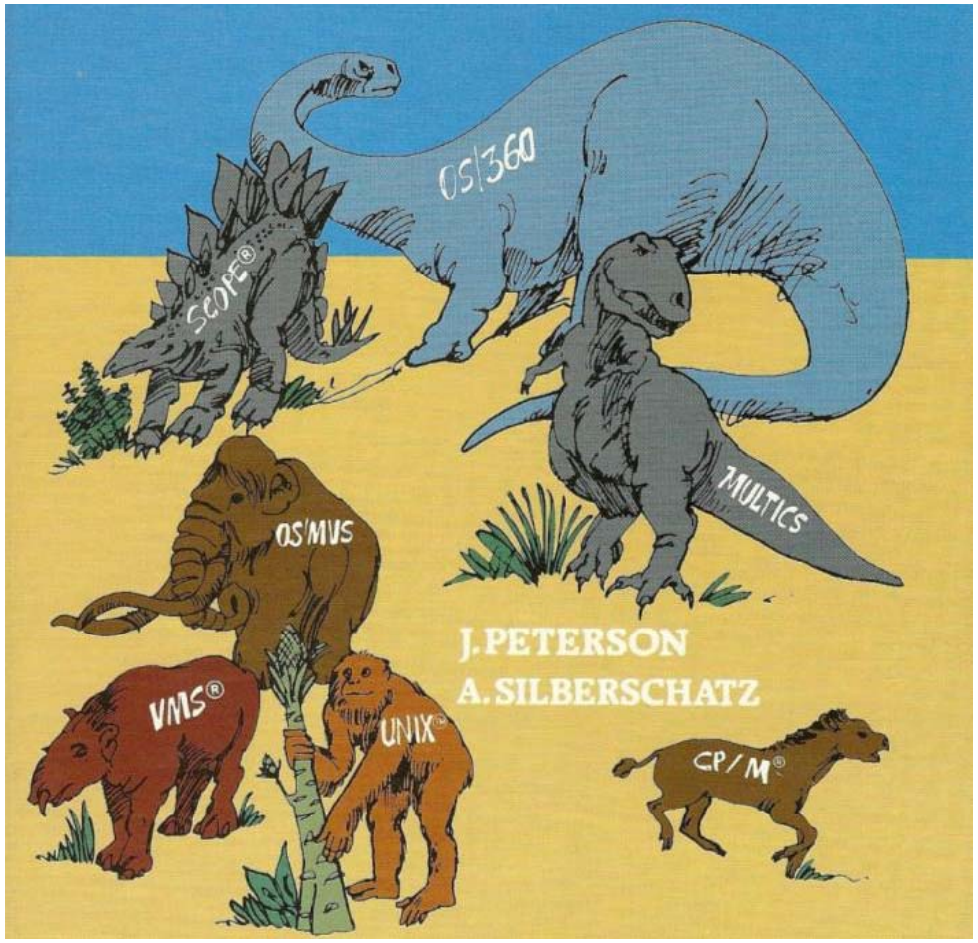
Eerst: geloof in evolutie & vooruitgang

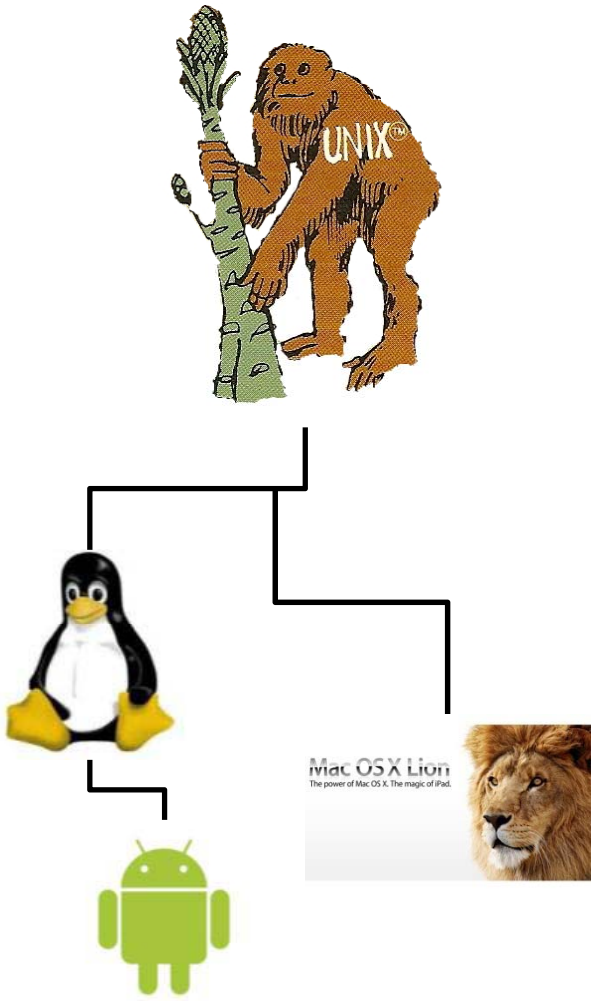


Eerste editie

Source: http://www.galvin.info/history-of-operating-system-concepts-textbook

Eerst: geloof in evolutie & vooruitgang



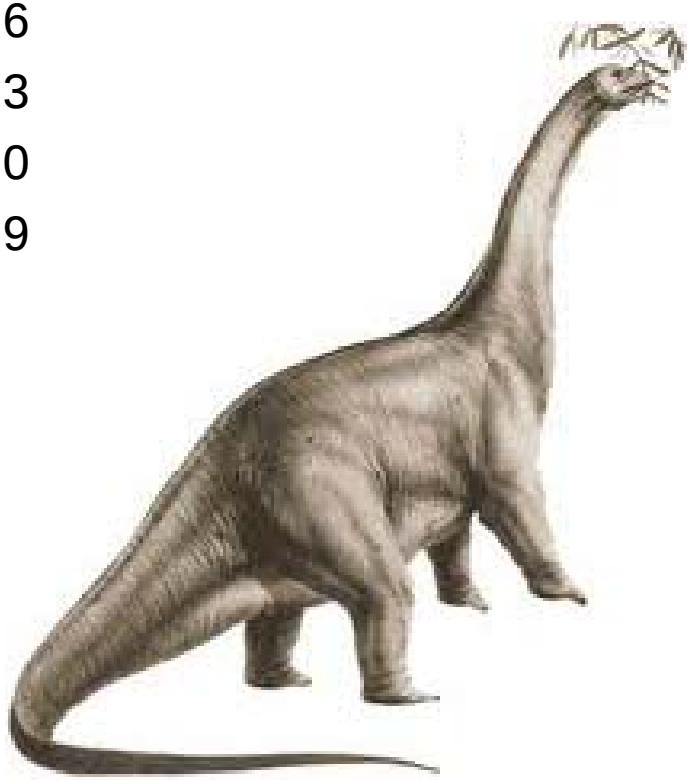


Eerste editie

Source: http://www.galvin.info/history-of-operating-system-concepts-textbook

Of produceert evolutie steeds dinosauriërs?

- UNIX 1971 33 system calls
- UNIX 1979 47
- SunOS 4.1 1989 171
- 4.3 BSD 1991 136
- HP UX 95. 1994 163
- SunOS 5.6 1997 190
- Linux 2.0 1998 229



Het einde van de dinosauriërs?

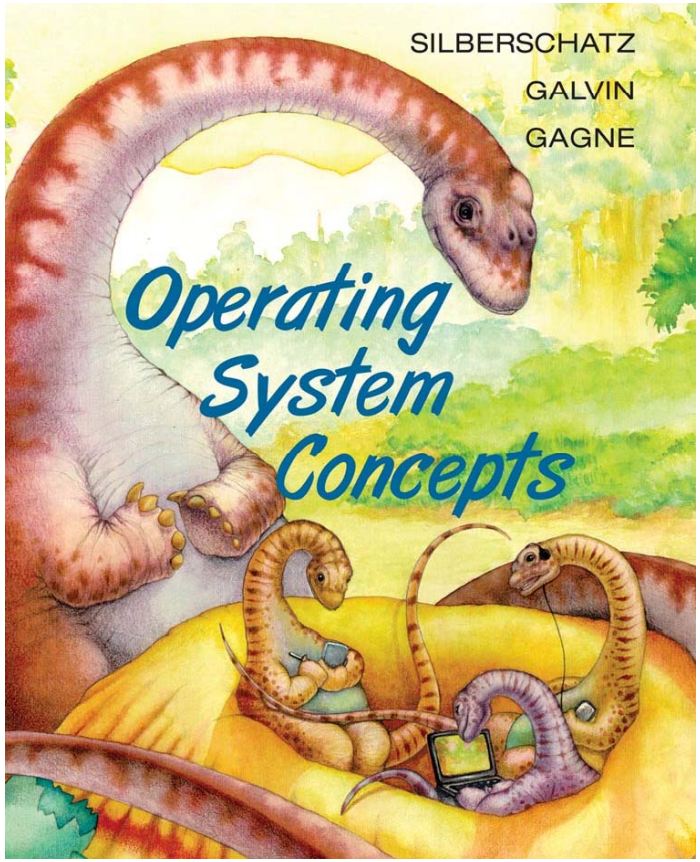
MS-DOS

Heel simpel OS,

voor disk management

Maar MS-DOS evolueerde ook

• UNIX 1971	33 system calls
• UNIX 1979	47
• SunOS 4.1 1989	171
• 4.3 BSD 1991	136
• HP UX 95. 1994	163
• SunOS 5.6 1997	190
• Linux 2.0 1998	229
• Windows NT 1999	3433
• …	..

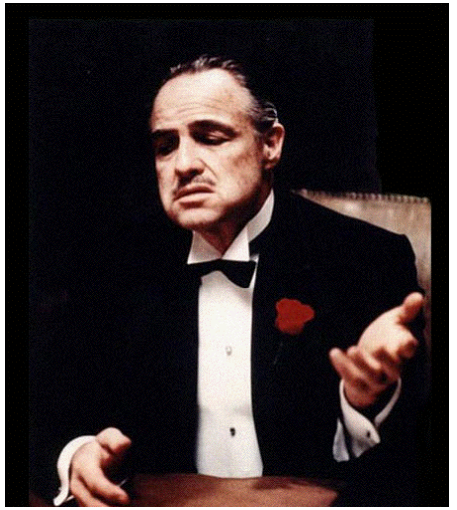


Alle computers evolueren naar vergelijkbare complexe operating systems

?

De evolutie van de aanvaller

hacker,1983



hacker, 2005

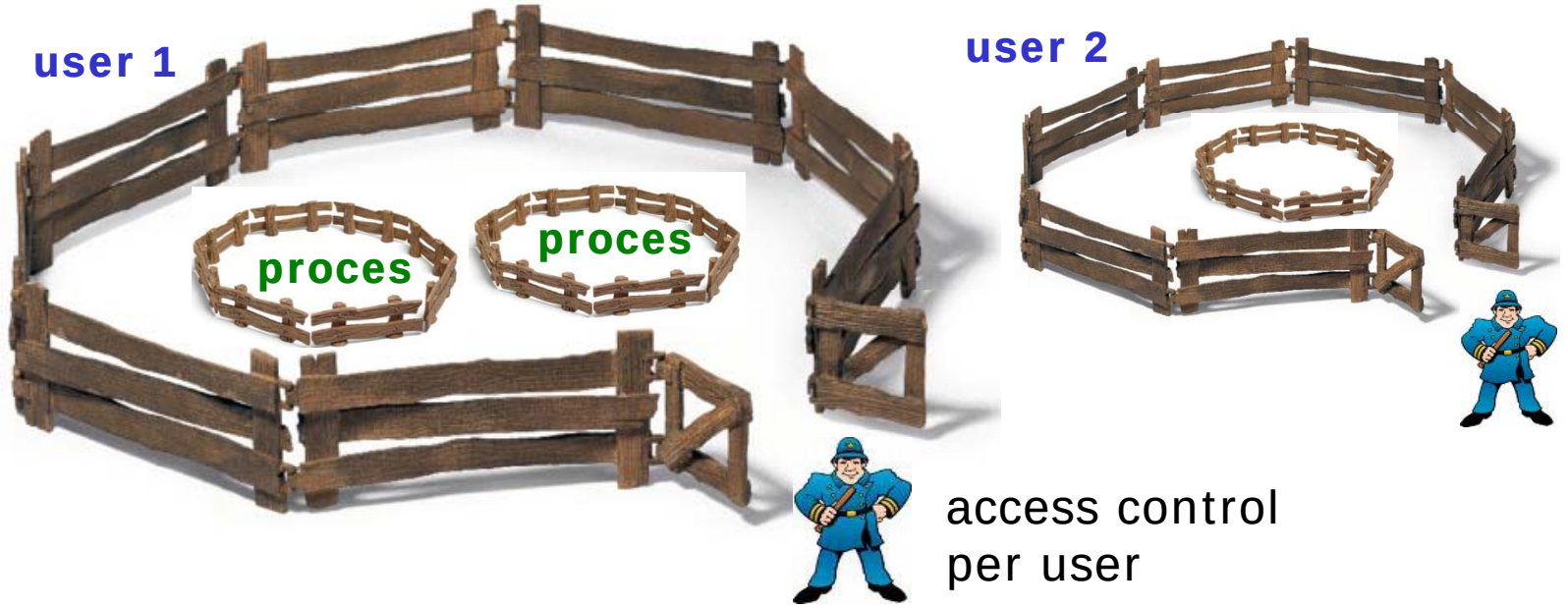
bijv:
fraude
internet
bankieren:
36Meuro
in 2012

bijv:
StuxNet

hacker, 2010

OS beveiliging: separatie & access control

- isolatie/separatie van processen
- access control per gebruiker



Moderne variaties hierop

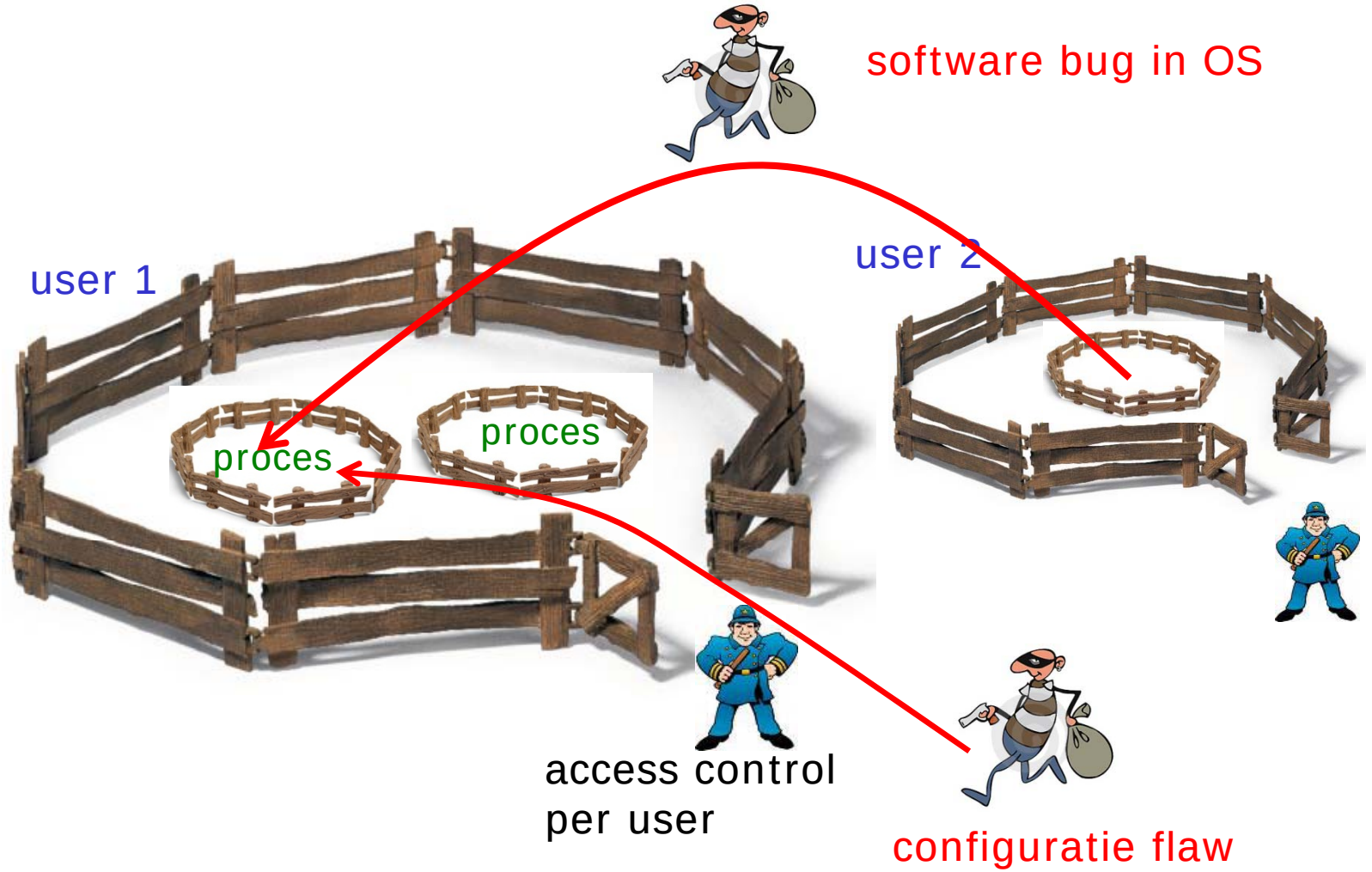
- Android phone heeft maar één gebruiker
  - aparte user voor elke app, voor fijnkorreligere access control

- Web-browser is één (groot & complex) proces
  - als je alles in de cloud doet, heb je dan nog wel een OS nodig? Is een browser niet genoeg?

IE8 en Chrome creëren een apart proces per tab

- voor fijnkorrelige access control
- om te voorkomen dat kwaardaardige spul in één tab de andere kan beinvloeden

Twee soorten OS beveiligingsgaten



Twee soorten OS beveiligingsgaten

1. *Bugs* in de enorm hoeveelheid code
   - *OS doet niet wat het moet doen*
   - *bijv. door buffer overflow*

2. *Flaws* in het ingewikkelde gebruik
   - *OS doet wat er gevraagd wordt, maar wat er gevraagd wordt (de configuratie) is onveilig*

In beide gevallen: wortel van alle kwaad is *complexiteit*.

example objects & permissions in Windows

Files and directories

Named pipes

Anonymous pipes

Processes Threads

File-mapping objects

Access tokens

Window-management objects
(window stations and desktops)

Local or remote printers

Network shares

Interprocess synchronization objects
(events, mutexes, semaphores, and waitable timers)

Job objects

GENERIC_READ

GENERIC_WRITE

GENERIC_EXECUTE

SC_MANAGER_ALL_ACCESS

SC_MANAGER_CREATE_SERVICE

SC_MANAGER_CONNECT

SC_MANAGER_ENUMERATE_SERVICE

SC_MANAGER_LOCK

SC_MANAGER_MODIFY_BOOT_CONFIG

SC_MANAGER_QUERY_LOCK_STATUS

SERVICE_ALL_ACCESS

SERVICE_CHANGE_CONFIG

SERVICE_ENUMERATE_DEPENDENTS

SERVICE_INTERROGATE

SERVICE_PAUSE_CONTINUE

SERVICE_QUERY_CONFIG

SERVICE_QUERY_STATUS

SERVICE_START

SERVICE_STOP

SERVICE_USER_DEFINED_CONTROL

ACCESS_SYSTEM_SECURITY

DELETE

READ_CONTROL

WRITE_DAC

WRITE_OWNER

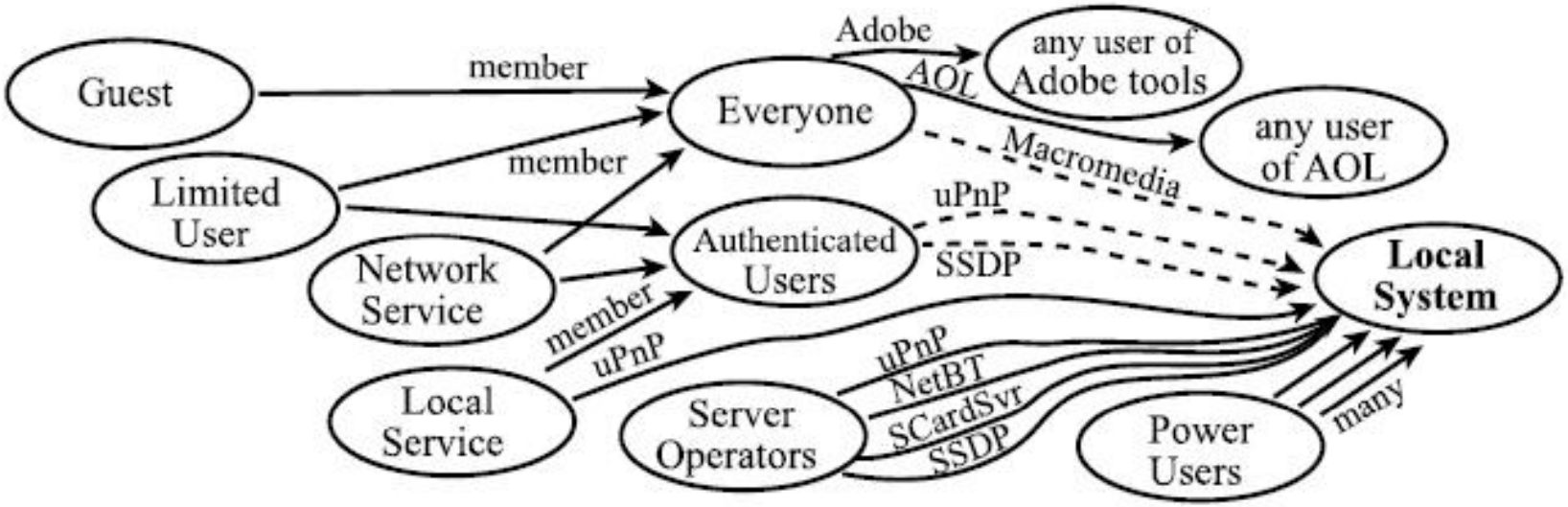
Voorbeeld: Complexiteit in OS gebruik

Beruchte privilige escalation in Windows

- Privileged functionality te realiseren als service, die onder bijv. Local Service or Local System account draaien
- Met SERVICE_CHANGE_CONFIG permissie kan iemand *de executable wijzigen* die bij een service hoort wijzigen (bijv. een printer driver kiezen)
- Maar met deze permissie kan je ook *de account wijzigen* waaronder de service draait wijzigen
  - bijv in Local System, wat maximale rechten geeft
- Services geven soms SERVICE_CHANGE_CONFIG aan alle Authenticated Users... Oeps.

privilege escalation in Windows XP

met standaard software van bekende vendors…



["Windows Access Control Demystified" by S. Govindavajhala and A.W. Appel]

Fundamenteel probleem

- steeds verfijndere toegangscontrole,
  om precies die rechten te geven die nodig zijn
  (principle of least privilige)

vs

- hou het simpel & begrijpelijk
  (KISS)

Hoe krijgen we iets betrouwbaar(der)s?

- hardening
  - verbeteren van configuratie, en verkleinen aanvalsdoelwit

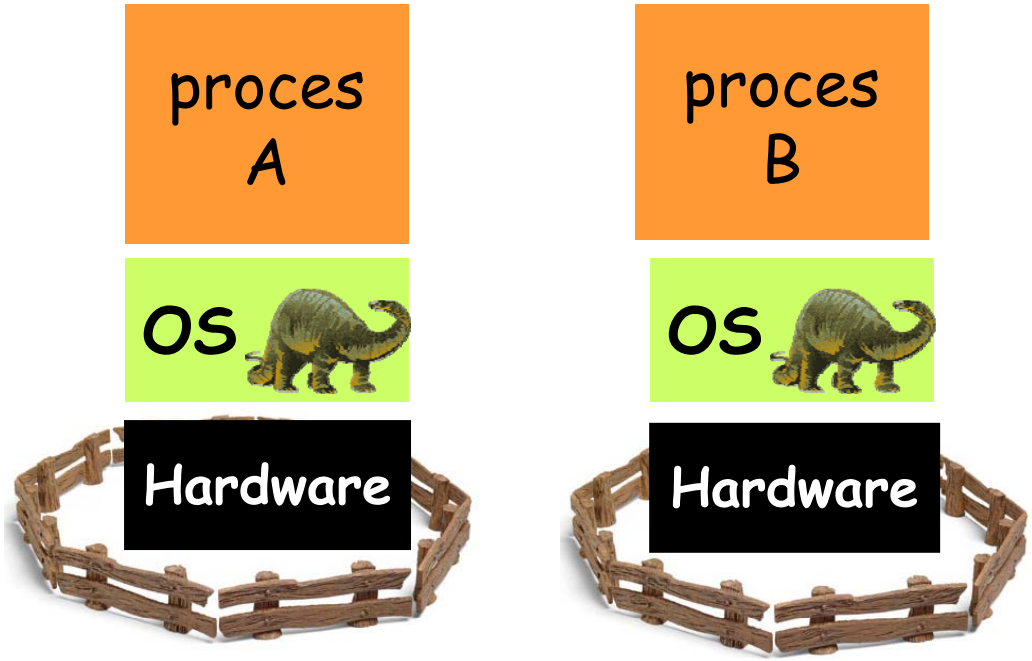
- microkernel
  - isoleer de security-kritische delen van het OS in een relatief kleine (micro)kernel

- geef het op ...

  om het OS betrouwbaar te krijgen

  maar introduceer een extra beveiligingslaag,
  - voor betrouwbare en simpele separatie

1. Verschillende fysieke machines



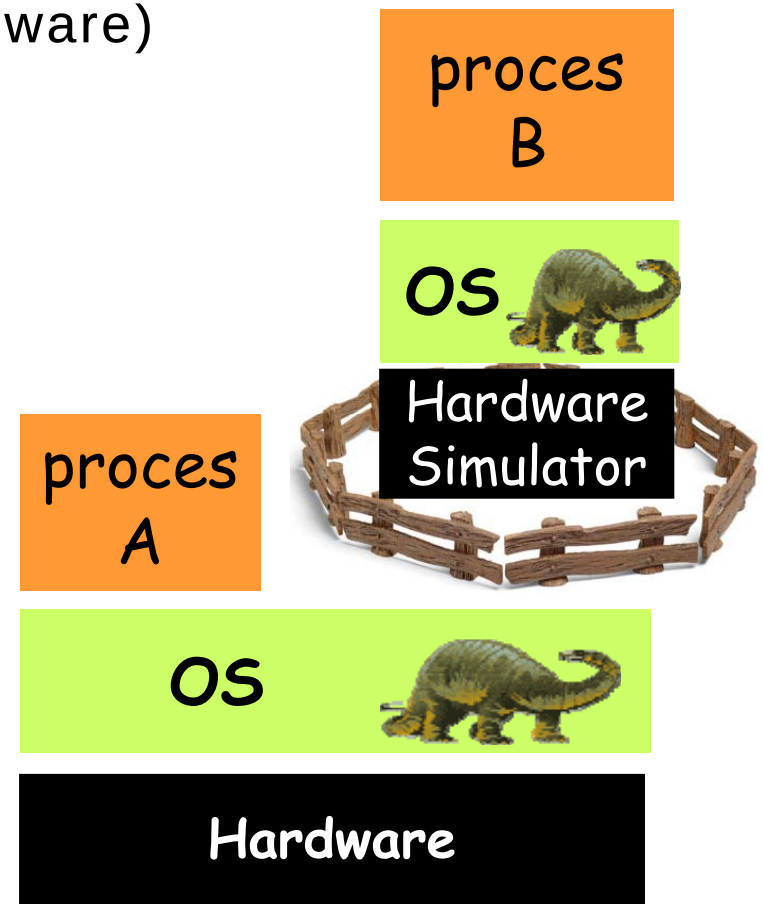
Populair in militaire toepassingen:

- MILS (Multiple Independent Levels of Security)

2. Virtualisatie met virtual machine

Virtual Machine (bijv van VMware)
simuleert hardware

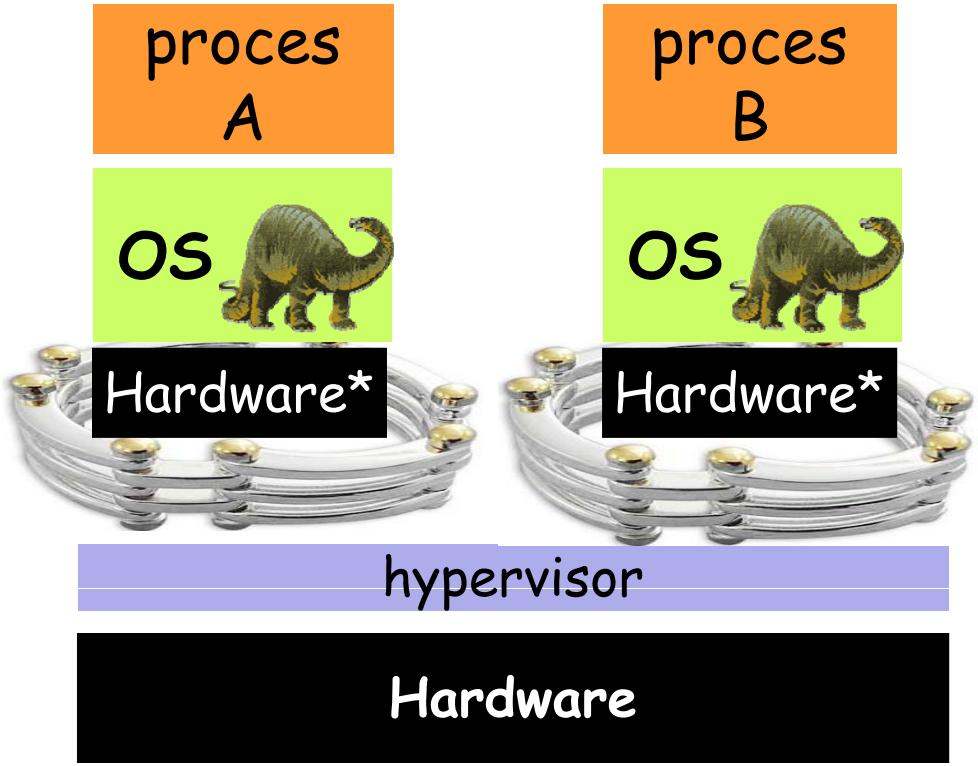
Populair in de cloud,
maar voor gemak en
niet voor betere
beveiliging!



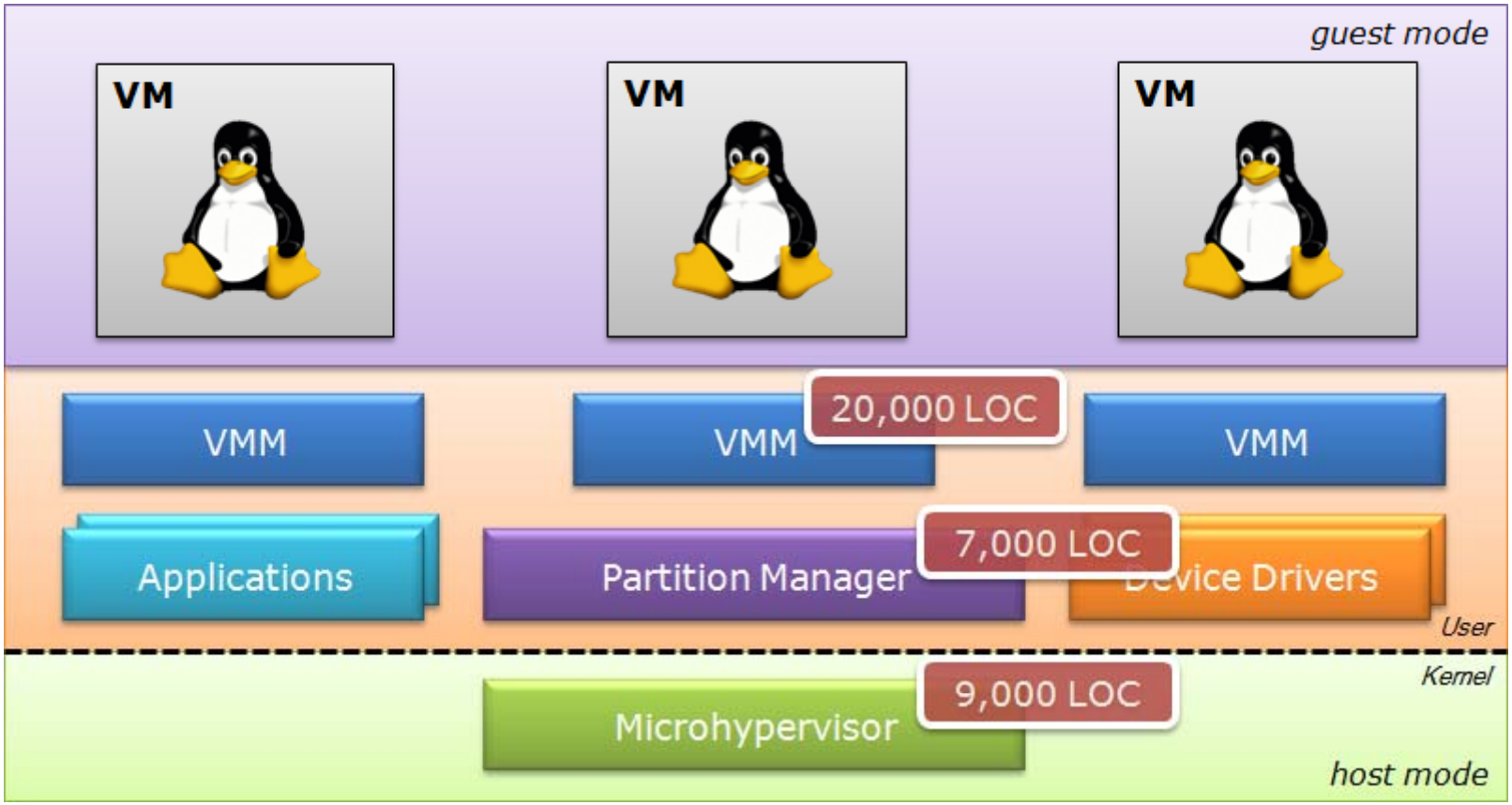
3. Virtualisation met hypervisor

Dun software laagje dat hardware repliceert

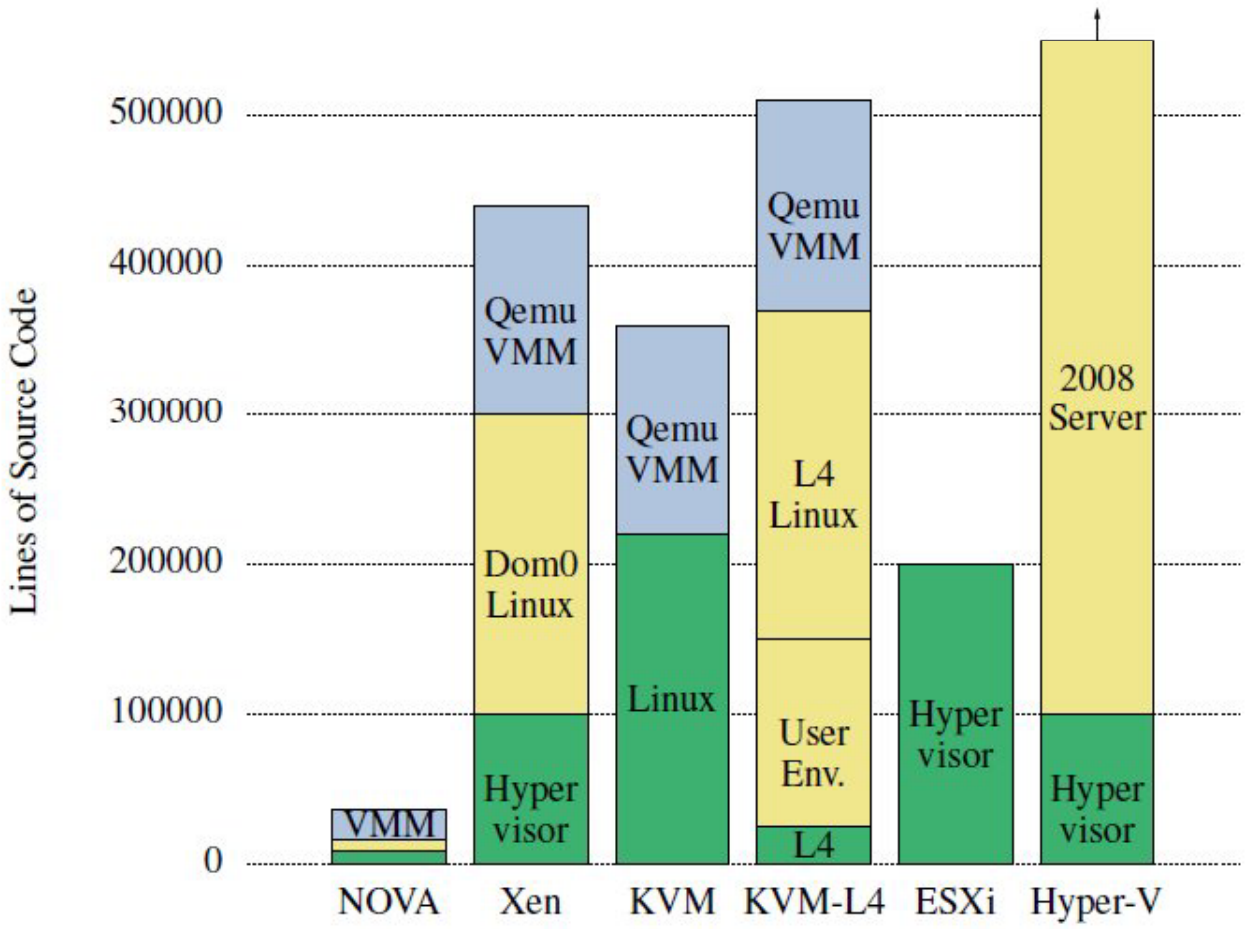
- net als VMware, maar draait onder OS, op de rauwe hardware



Moderne *micro*hypervisor: NOVA [TU Dresden]



Vergelijking TCB (Trusted Computing Base)



Een trustworthy Trusted Computing Base?

Met Formele methoden!

- Microsoft's Hyper-V hypervisor geverifiëerd met VCC
  - 60 kloc C en 4.5kloc x86 assembly
  - correct = elk programma gedraagt zich identiek als het op de hypervisor draait of op de hardware
- seL4 kernel geverifiëerd met Isabelle/HOL
  - 9 kloc C en .5 kloc assembly
  - correct = C code gedraagt zich net als Haskell prototype

Het goede nieuws: dit kan!

Het slechte nieuws: het kost nog te veel tijd, geld, en moeite...

Conclusies

- beveiligingszwakheden
  - *in* operating systems, en
  - *in het gebruik* van operating systems

  zullen blijven bestaan,

  en in steeds meer vormen van computers voorkomen

- hypervisors bieden een mogelijkheid om toch nog betrouwbaar eenvoudige beveiliging te krijgen

- *Hebben we straks hypervisors op onze smartphones om te geld over te maken? of om te bellen?*

Vragen?