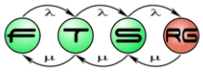
Intelligens rendszerfelügyelet

Szerveroldali virtualizáció

Tóth Dániel, Micskei Zoltán



Budapesti Műszaki és Gazdaságtudományi Egyetem
Méréstechnika és Információs Rendszerek Tanszék

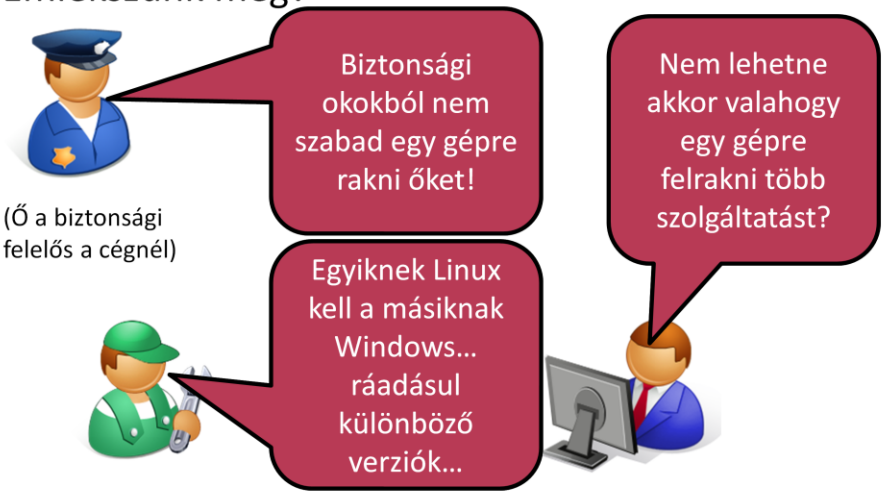
Utolsó módosítás: 2012. 04. 17.

Motivációs példa

Motivációs példa

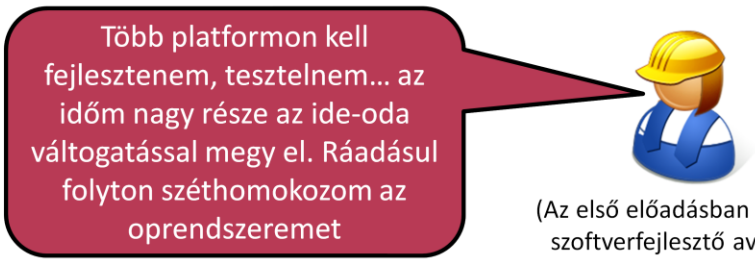
Motivációs példa

- Emlékszünk még?

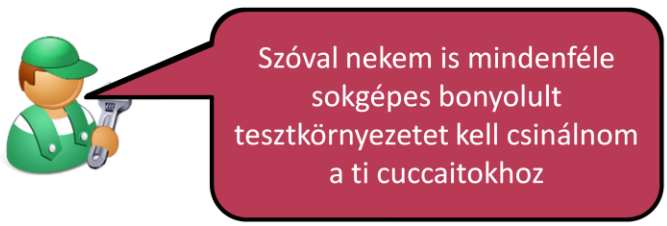


Motivációs példa

- „Now for something completely different…”



(Az első előadásban ő volt a szoftverfejlesztő avatarja)



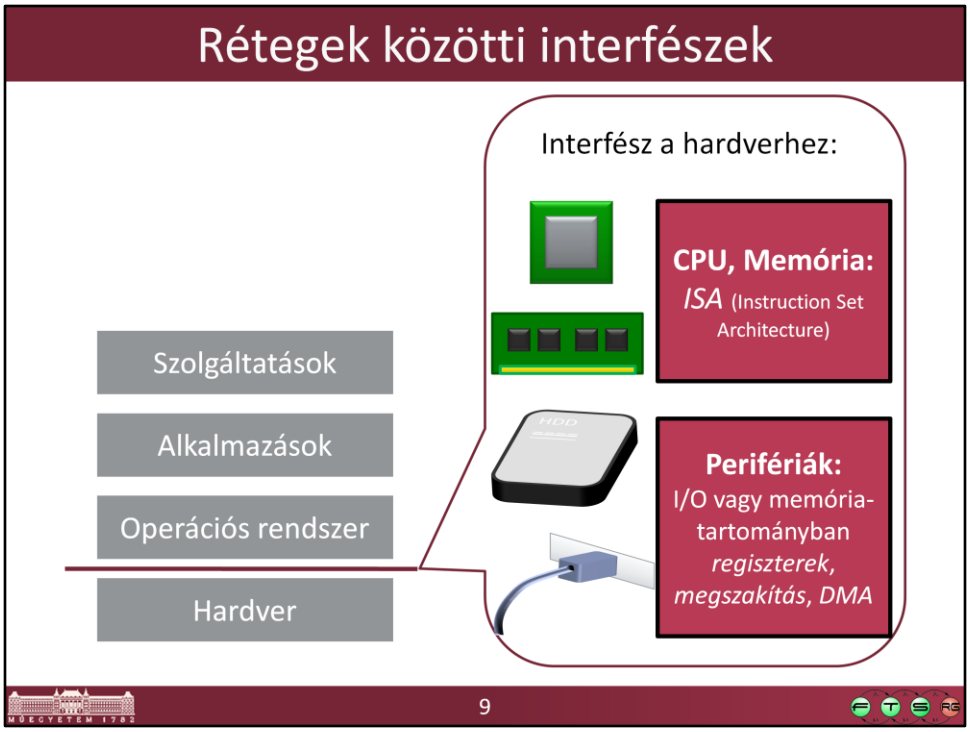
Virtualizáció

- Mi az a virtualizáció?
- „Az erőforrások elvonatkoztatása az erőforrást nyújtó elemektől”
  - kellemesen sejtelmes általános definíció ☺
- Jellemzően:
  - fizikai erőforrásokból logikai erőforrások képzése, amik függetlenek a tényleges fizikai elemektől
  - korlátos erőforrások szétosztása több részre
- Ez egy új ötlet?
  - Korántsem – az oprendszerek is ezt csinálják…

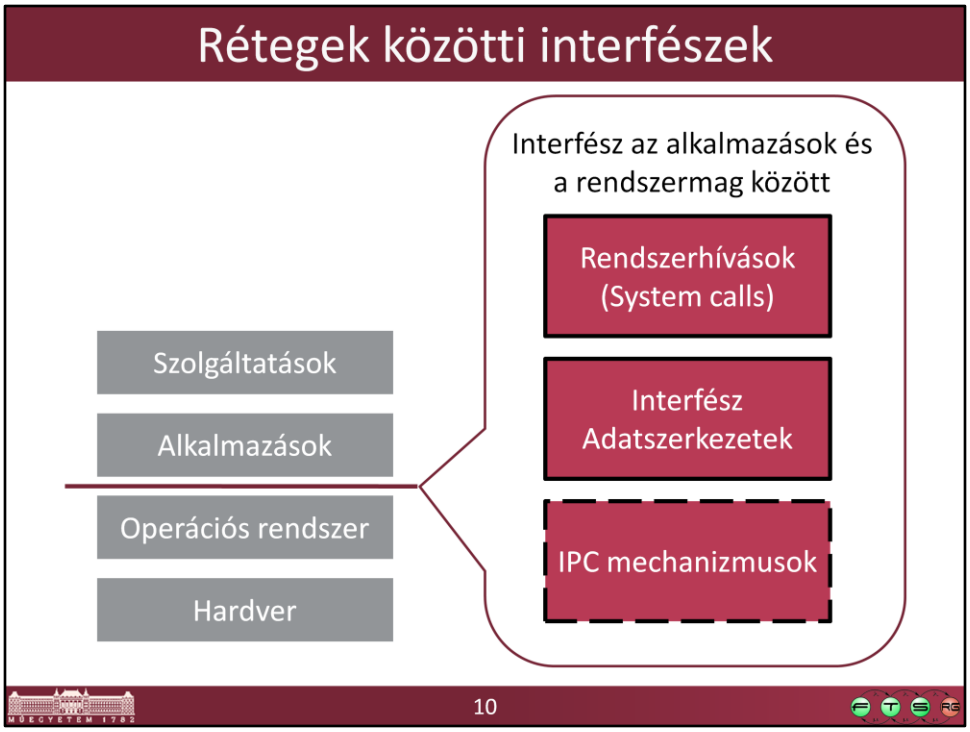
Tartalom

- **Ismétlés (lásd Operációs rendszerek)**
  - Virtualizáció fajtái
  - Platform virtualizációs megoldások
- Szerveroldali virtualizáció

Erősen buzzword-fertőzött terület, manapság mindent szeretnek „virtualizáció” fogalmai alá berakni.

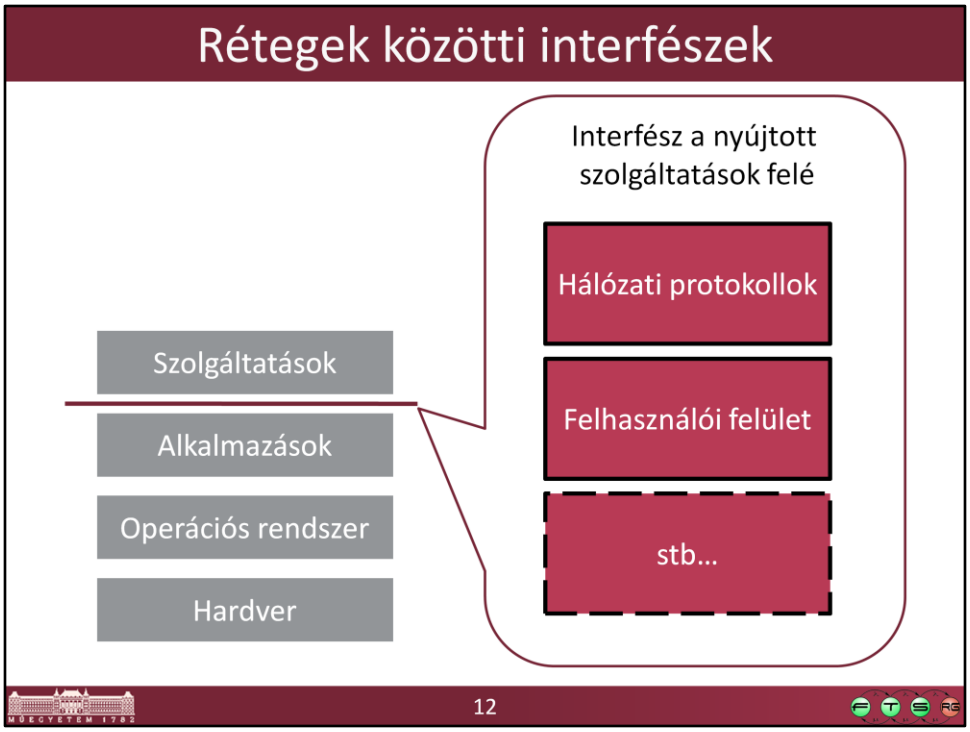


„Platform virtualizáció”
Itt a lényeg, hogy a virtuális gépekben egy-egy saját kernel fut valamilyen izolált környezetben, tehát egymástól eltérő fajta operációs rendszerek is futhatnak egymás mellett. Általában (de nem minden esetben, lásd. paravirtualizáció) a virtuális gépben futó OS számára egy látszólag valódi fizikai hardverhez hasonló hardverkörnyezetet biztosít.



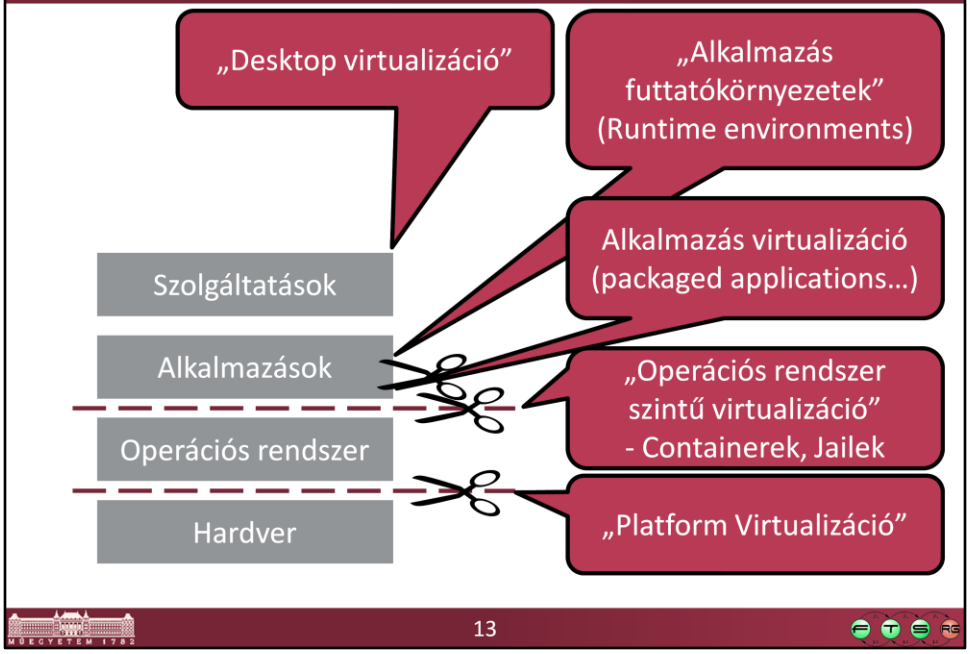
„Operációs rendszer szintű virtualizáció” vagy „Konténer alapú virtualizáció”
Itt az operációs rendszer felett alakítunk ki elkülönített virtuális környezeteket (jail, container), amely közül mindegyik úgy érzékeli, hogy csak ő fut a kernel felett. Ehhez a kernel által biztosított – normális esetben singleton – erőforrásokat kell többszörözni minden környezethez. A megoldás változó mélységű lehet, van olyan, ahol csak a kernel látszik, van, olyan, ahol valamilyen közös magasszintű felhasználói-módú erőforrásokat is elérhetővé tesz minden izolált környezetben.

Igazából hagyományosan az OS-hez soroljuk, de nem a kernel, hanem felhasználói módban futó könyvtárak, folyamatok szolgáltatják az adott operációs rendszer magas szintű programkörnyezetét.
Az „alkalmazás szintű virtualizáció” igazából a konténer alapú virtualizációk egy olyan esete, ahol a használati eset specifikusan csak egy-egy alkalmazás „becsomagolása”, az OS környezet összes többi része nincs izolált konténerben. Pl. MS Office használata telepítés, registry módosítás, stb. nélkül.



„Desktop virtualizáció”-nak szokták hívni azt az esetet, amikor a felhasználó számára nyújtott szolgáltatást (elsődlegesen grafikus felületen) távolítjuk el szolgáltatási igénybevételi ponttól. Jellemző eset, hogy a felhasználó desktop környezete valami szervergépen fut (akár valamilyen más, platform vagy OS szintű virtualizációs formával megvalósítva), a kezelőfelület távoli elérésére azonban vékonykliens gépeket használnak.
Más esetben a szolgáltatás valamilyen hálózaton elérhető technikai szolgáltatás, webes felület stb. A manapság oly divatos „cloud computing” alapötlete, hogy a hálózati szolgáltatásoknál könnyen elfedhető, hogy valójában mely komponens biztosítja, ezt is egyfajta „virtualizációnak” szokták nevezni.

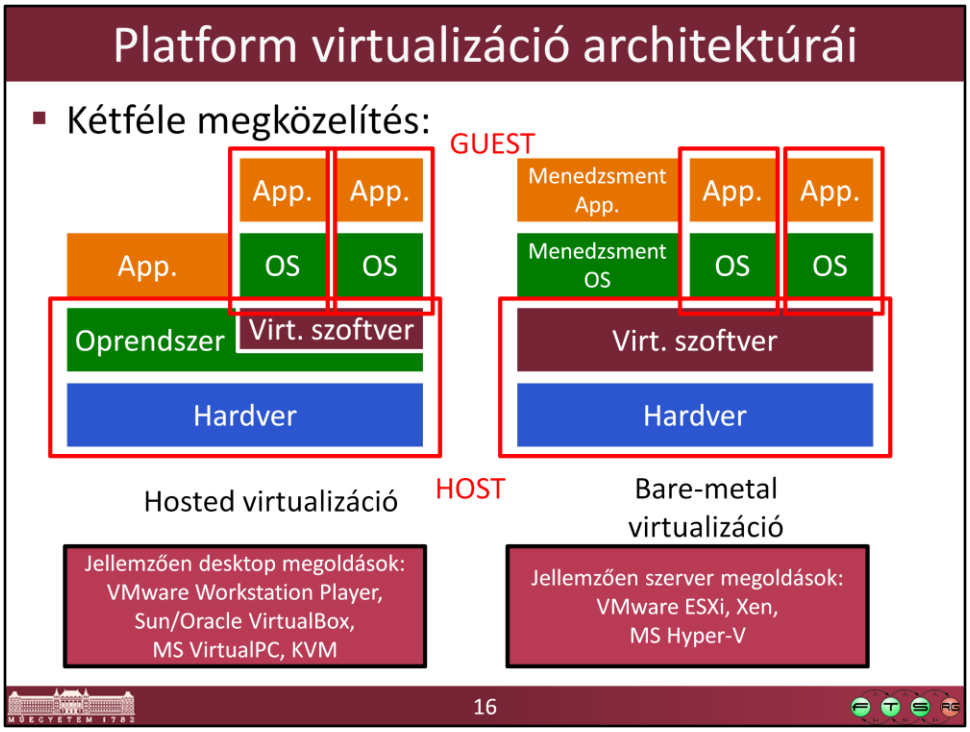
A virtualizáció különböző fajtái



Platform virtualizáció

- Amikor a „virtualizáció” buzzword elhangzik leggyakrabban erről van szó
  - *„Szerver virtualizáció”*, *„Hardver virtualizáció”*, *„Számítógép virtualizáció”* szinonim fogalmak
  - De nem összekeverendő a *„hardveres”* virtualizációval!
- Cél: megosztani a hardver erőforrásokat:
  - Nem végzünk finomítást, az eredeti(hez hasonló) interfészen maradnak elérhetőek
  - Izolált környezeteket („sandbox”) biztosítunk
- Célok gyakorlatiasabban megfogalmazva:
  - Több operációs rendszer példányt futtatni egyazon gépen

- A hardveres virtualizáció csak egy lehetséges technika a platform virtualizáció megvalósítására, kb. rész-egész viszonyban vannak. Lásd később…
- A hardver erőforrások megosztásának fogalmába általában belefér az, hogy nem pont azonos interfészen osztjuk meg az erőforrást, tehát lehet más CPU architektúra, más típusú periféria, de fontos megkülönböztetés, hogy nem képezünk belőle magasszintű logikai szolgáltatásokat, mint egy hagyományos OS teszi. Persze ez alól is lesznek kivételek, de nagyvonalakban ez igaz marad.



- **FONTOS**: Bare-metal esetén a VMM kezeli a HW erőforrásokat, míg hosted típusú esetén ezt a host OS végzi.
- Azt a megkülönböztetést érdemes kerülni, hogy a VMM a hardveren fut vagy hogy hosted esetben van a gépen operációs rendszer is (mert tulajdonképpen base-metal esetben a virtualizációs szoftver is egy speciális operációs rendszer).
- A Bare-metal esetben szokták a VMM-et hypervisornak nevezni, de itt is szoktak eltérések lenni az elnevezésben.
- Szokták még a Type I és Type II megkülönböztetést is használni, de ez nem annyira egyértelmű, a hosted és bare-metal felosztás szerintem szerencsésebb (lásd: Micskeiz Zoltán. Type I vs. Type II VMM. URL: https://micskeiz.wordpress.com/2011/11/13/type-i-vs-type-ii-vmm/).

Platform virtualizáció

Operációs rendszerekből érdemes átismételni

- Tiszta emuláció
  - Miben különbözik a virtualizációtól, hol kerül elő a virtualizáción belül
- Hogy is működik?
  - Elfogás és emuláció elve (trap & emulate)
- Virtualizáció
  - Szoftveres virtualizáció (bináris átírás)
  - Paravirtualizáció
  - Hardveres virtualizáció (trap & emulate, teljesen hardveres támogatással)

Tartalom

- Ismétlés (lásd Operációs rendszerek tantárgy)
  - Virtualizáció fajtái
  - Platform virtualizációs megoldások
- **Szerveroldali virtualizáció**
  - Platform virtualizációs megoldások
  - Erőforrás-gazdálkodás
  - Operációs rendszer szintű virtualizáció

Szerver virtualizáció

- Jellegzetességek
  - Távoli elérés központi szerepe
  - Cél.: Hálózaton nyújtott szolgáltatások ellátása
    (ez akár távoli asztal is lehet! -> Desktop virtualizáció)
  - Erőforrás gazdálkodás
  - Központi menedzsment fontossága
    (következő előadás…)
- Kétféle megoldás(t tárgyalunk most)
  - Platform virtualización alapuló
  - Operációs rendszer szintű virtualizáción alapuló

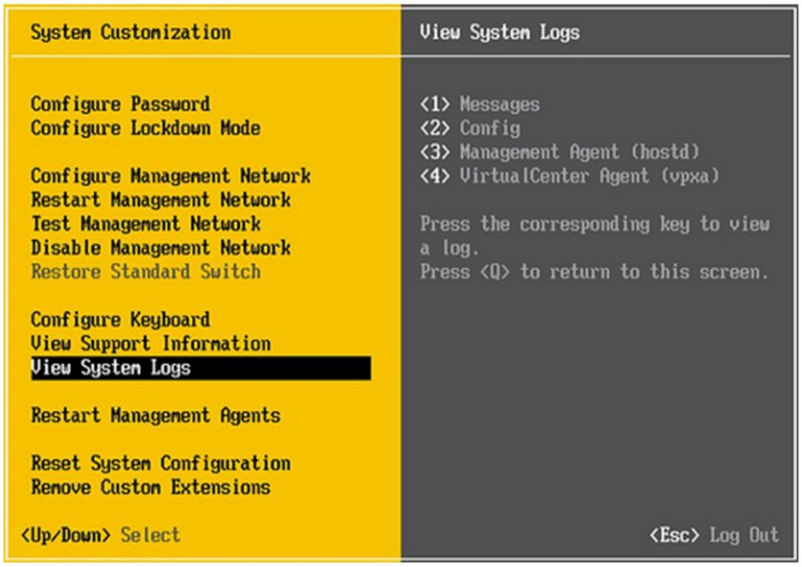
Szerver virtualizáció

- Platform virtualizáción alapuló megoldások:
  - VMware vSphere (ESXi)
  - Xen (+ Citrix XenServer, Oracle VM…)
  - Microsoft Hyper-V
  - IBM LPAR, DLPAR
- Operációs rendszer szintű virtualizáción alapul:
  - Linux OpenVZ
  - Linux VServer
  - Parallels Virtuozzo Containers (Linux, Windows)
  - Solaris Containers, Zones
  - FreeBSD jail
  - AIX WPAR

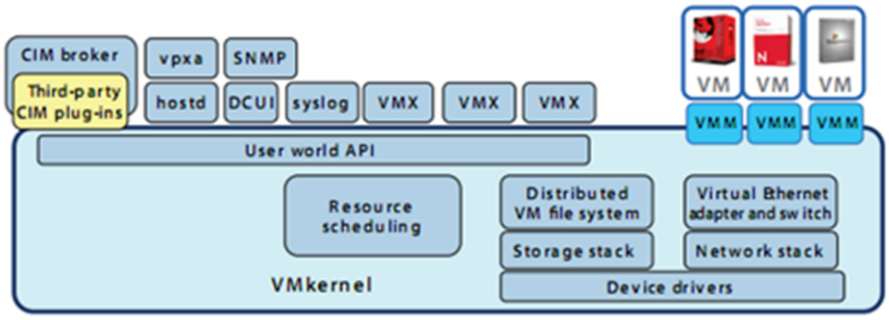
VMware ESXi

- VMware bare-metal megoldása
  - Követelmény: 64 bites CPU
  - (Van ingyenes verziója is)
- VMware ESX utódja (új architektúra)
- vSphere 5 család része
  - vCenter Server, vMotion, DRS, HA, FT…
  - lásd a következő előadást

VMware ESXi: Minimális konzol

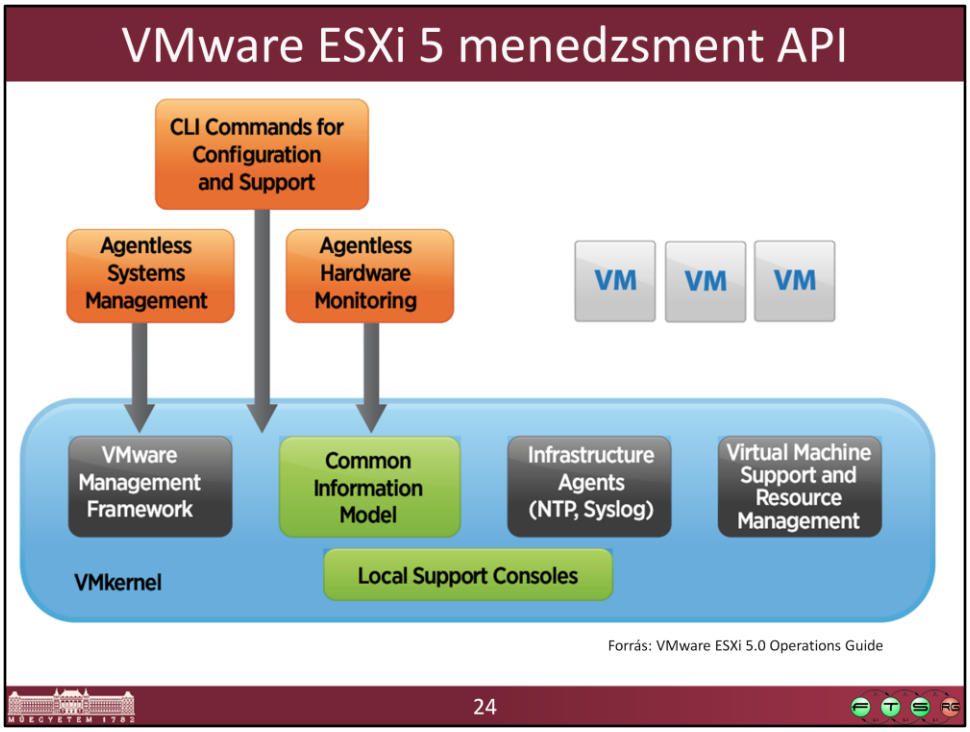


VMware ESXi 5 architektúrája



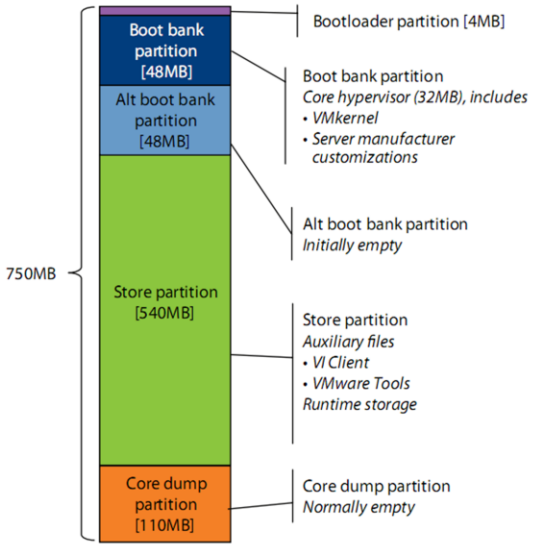
- VMkernel
- World
  - VM world
  - Ágensek, shell…

- Az ábrán látszik részletesebben, hogy milyen funkciókat kell a VMkernelnek ellátnia.
- A VMkernel úgynevezett worldöket futtat (~ ez felel meg kb. a folyamatnak). Ezek lehetnek ágensek, helyi shell, menedzsment eszközök vagy pedig virtuális gépek és segédfolyamataik.



- Forrás: VMware, VMware ESXi 5.0 Operations Guide**,** Technical white paper, 2011. URL: http://www.vmware.com/resources/techresources/10215
- Mendzsment APIk
  - helyi és távoli parancssoros felület (CLI)
  - Web Service alapú API programozott eléréshez (Java, .NET, PowerShell…)
  - CIM alapú HW felügyelet

ESXi system image



- Aktív és alternatív verzió
- In-memory fájlrendszer
  - Pl. log fájl elveszik rebootkor
- Egyszerű frissítés
- OEM kiegészítések (embedded ESXi)

Forrás: VMware, The Architecture of VMware ESXi

Fontos ötletek:

- Dual boot bank: hibás frissítés esetén csak újra kell indítani, és visszaállhatunk a régire
- Egyben frissíthető az egész, nem kell frissítések függőségeit keresni, „firmware”-szerűen cserélhető az egész
- Ehhez kell az in-memory fájlrendszer is, a konkrét beállítások egy state.tgz fájlban tárolódnak.

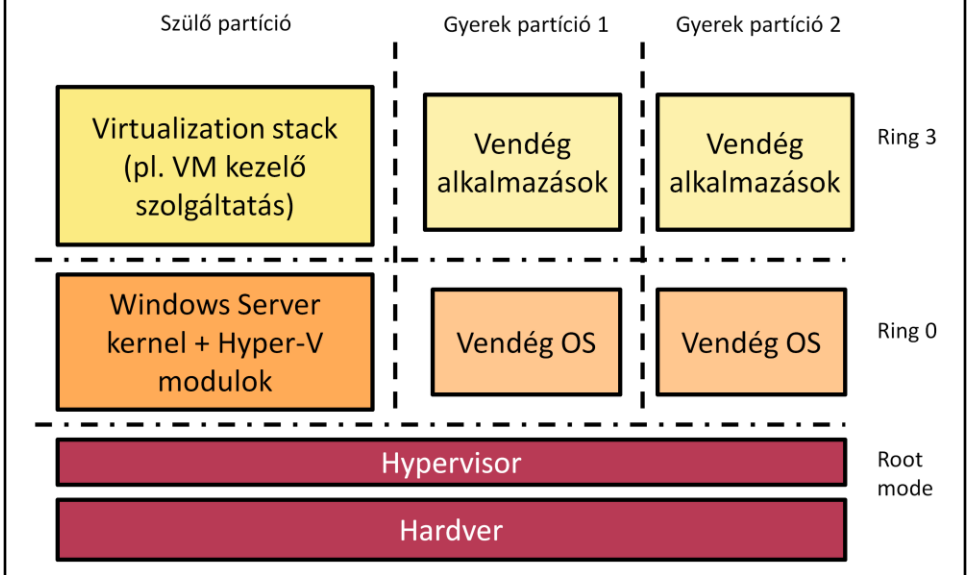
Források:

- Kép forrása: VMware, The Architecture of VMware ESXi, White Paper, URL: http://www.vmware.com/resources/techresources/1009
- Lásd még: http://micskeiz.wordpress.com/2010/11/25/esxi-4-1-felptse-s-partcii/
- Technikai részletek: Olivier Cremel. VisorFS: A Special-purpose File System for Efficient Handling of System Images. VMware Technical Journal, Issue 1, 2012. URL: http://labs.vmware.com/publications/cremel-vmtj-spring2012

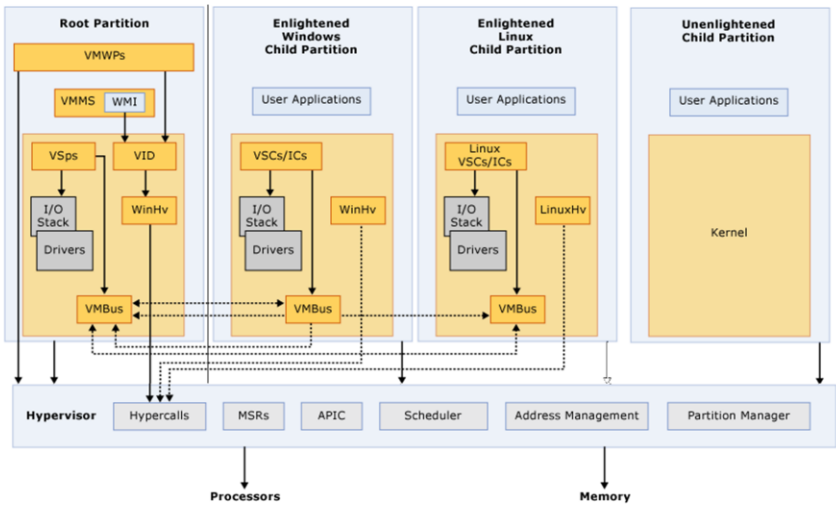
Microsoft Hyper-V

- Microsoft bare-metal virtualizációs megoldása
  - Jelenleg: 2. verzió (R2) / 3. verzió béta (WS 8)
- Két változat:
  - Windows Serverben a Hyper-V szerep
  - MS Hyper-V Server (különálló, ingyenes, csak Hyper-V)
- HW igény:
  - CPU: 64 bites, HW-es virtualizációs támogatás

Hyper-V architektúra (1)



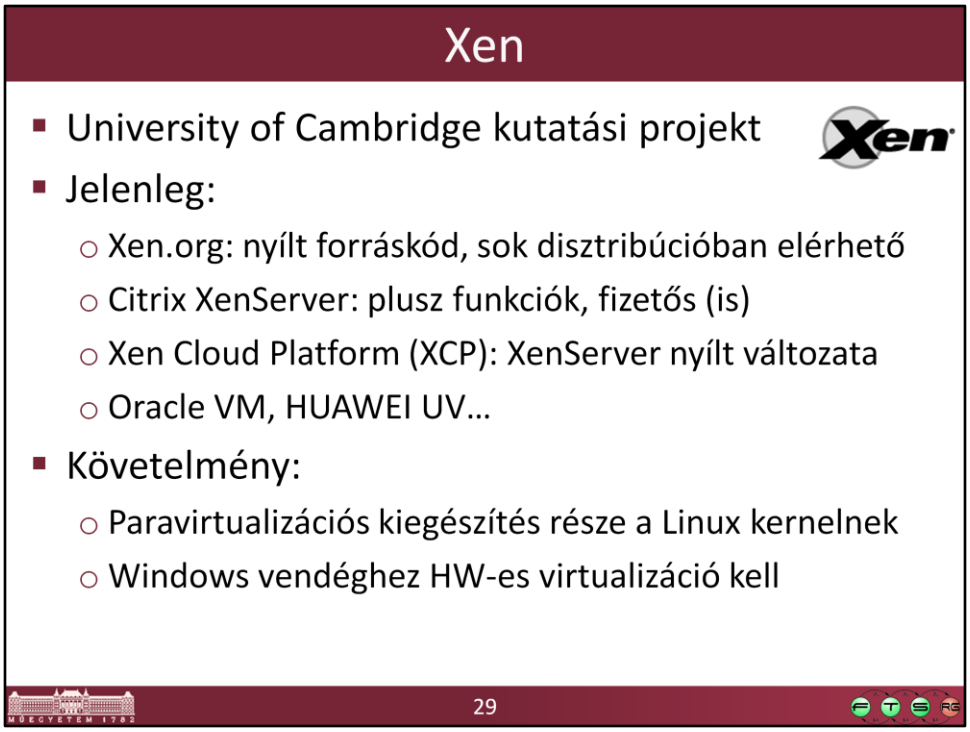
Hyper-V architektúra (2)



Forrás: Microsoft. Hyper-V Architecture and Feature Overview

Forrás: Appendix B: Hyper-V Architecture and Feature Overview
http://msdn.microsoft.com/en-us/library/dd722833(BTS.10).aspx

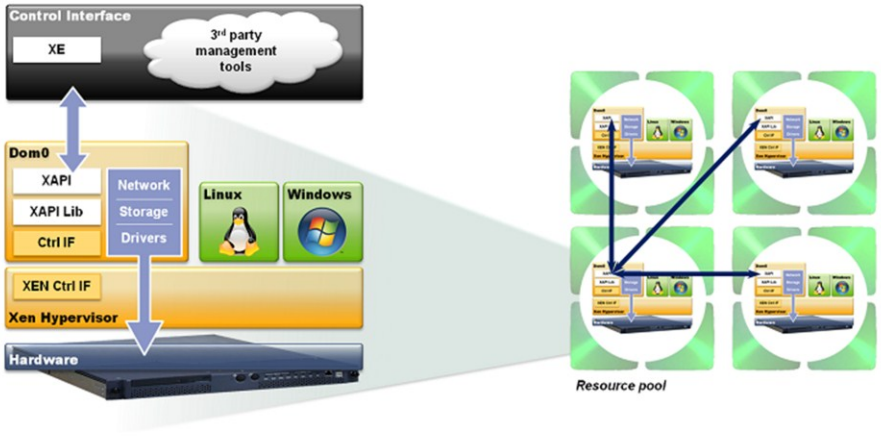
„Acronyms and terms used in the diagram above are described below:
• **APIC** – Advanced Programmable Interrupt Controller. A device which allows priority levels to be assigned to its interrupt outputs.
• **Child Partition** – Partition that hosts a guest operating system. All access to physical memory and devices by a child partition is provided via the Virtual Machine Bus (VMBus) or the hypervisor.
• **Hypercall** – Interface for communication with the hypervisor. The hypercall interface accommodates access to the optimizations provided by the hypervisor.
• **Hypervisor** – A layer of software that sits between the hardware and one or more operating systems. Its primary job is to provide isolated execution environments called partitions. The hypervisor controls and arbitrates access to the underlying hardware.
• **IC** – Integration component. Component that allows child partitions to communication with other partitions and the hypervisor.
• **I/O stack** – Input/output stack.
• **MSR** – Memory Service Routine.
• **Root Partition** – Manages machine-level functions such as device drivers, power management, and device hot addition/removal. The root (or parent) partition is the only partition that has direct access to physical memory and devices.
• **VID** – Virtualization Infrastructure Driver. Provides partition management services, virtual processor management services, and memory management services for partitions.
• **VMBus** – Channel-based communication mechanism used for inter-partition communication and device enumeration on systems with multiple active virtualized partitions. The VMBus is installed with Hyper-V Integration Services.
• **VMMS** – Virtual Machine Management Service. Responsible for managing the state of all virtual machines in child partitions.
• **VMWP** – Virtual Machine Worker Process. A user mode component of the virtualization stack. The worker process provides virtual machine management services from the Windows Server 2008 instance in the parent partition to the guest operating systems in the child partitions. The Virtual Machine Management Service spawns a separate worker process for each running virtual machine.
• **VSC** – Virtualization Service Client. A synthetic device instance that resides in a child partition. VSCs utilize hardware resources that are provided by Virtualization Service Providers (VSPs) in the parent partition. They communicate with the corresponding VSPs in the parent partition over the VMBus to satisfy a child partitions device I/O requests.
• **VSP** – Virtualization Service Provider. Resides in the root partition and provide synthetic device support to child partitions over the Virtual Machine Bus (VMBus).
• **WinHv** – Windows Hypervisor Interface Library. WinHv is essentially a bridge between a partitioned operating system's drivers and the hypervisor which allows drivers to call the hypervisor using standard Windows calling conventions
• **WMI** – The Virtual Machine Management Service exposes a set of Windows Management Instrumentation (WMI)-based APIs for managing and controlling virtual machines."



- Lásd: Paul Barham et al. (2003) Xen and the art of virtualization. In *Proceedings of the nineteenth ACM symposium on Operating systems principles* (SOSP '03). ACM, New York, NY, USA, 164-177. URL: http://doi.acm.org/10.1145/945445.945462
- Wim Coekaerts Blog. Linux mainline contains all the Xen code bits for Dom0 and DomU support, May 26, 2011, URL: https://blogs.oracle.com/wim/entry/linux_mainline_contains_all_the

XCP architektúra

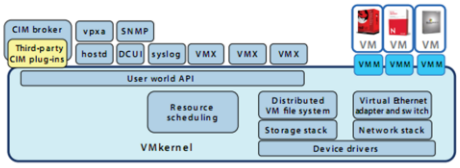
- Dom0: menedzsment OS
- DomU: virtuális gépek



Forrás: http://xen.org/products/cloudxen.html

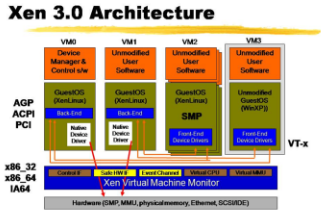
Bare metal megoldások architektúrái

ESXi



- I/O eszközöket is a hypervisor kezeli
- Meghajtókat a VMware szállítja
- Extra kis méret: ESXi (32 MB)

Xen / Hyper-V



- I/O eszközök kezelése a Dom0 / szülő partícióban
- Meghajtókat a HW gyártók szállítják

Tartalom

- Ismétlés (lásd Operációs rendszerek)
  - Virtualizáció fajtái
  - Platform virtualizációs megoldások
  - Kliens oldali virtualizációs igények
- **Szerveroldali virtualizáció**
  - Platform virtualizációs megoldások
  - **Erőforrás-gazdálkodás**
  - Operációs rendszer szintű virtualizáció

Erőforrás gazdálkodás

- A virtuális gépek közös erőforráson osztoznak
- Jellemző példák:
  - CPU: gyakran (összesen több vCPU, mint fizikai)
  - Memória: ritkábban (memory overcommit)
  - Háttértár I/O műveletek: itt jellegzetesen osztozás van!
  - Hálózati áteresztőképesség: itt is osztozás van

Versengés az erőforrásokért

- Erőforrás szűk keresztmetszet lesz
  - Kis terheléseknél ritka
  - De szerverkörnyezetben gyakran előfordul
- Hogyan osszuk el ilyenkor az erőforrásokat?

Feladatok erőforrások kezelésénél

- Tipikus igények:
  - Korlátozni valakinek a felhasználást
  - Garantálni valakinek a minimumot
  - Prioritás versenyhelyzet esetén
- Megoldások:
  - Kemény korlátozások, „lágy” korlátok, részesedés
  - „Proportional-Share Based Scheduler”

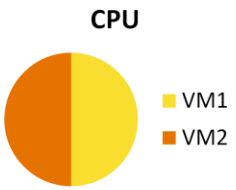
Szabályozási lehetőségek (VMware)

- **Resource Limit** – kemény felső korlát az erőforrás igénybevételére
  - Akkor is érvényes, ha egyébként van szabad erőforrás
- **Resource Reservation** – garantált rendelkezésre álló erőforrás mennyiség
  - Nem feltétlenül használja ki, csak verseny esetén érvényesül, egyébként a keretet más használhatja
- **Resource Shares** – prioritás
  - Verseny esetén az alapértelmezett „igazságos” elosztás módosítható ezzel

További információ (ütemező leírása, co-scheduling, „CPU topology aware load-balancing”, stb.):
VMware. „VMware vSphere: The CPU Scheduler in VMware ESX 4.1”, http://www.vmware.com/resources/techresources/10131

Példa a share használatára

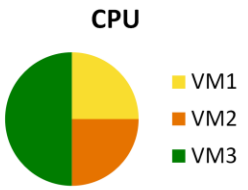
- Több VM-et futtató gép esetén a CPU share értékek a következők
  - VM1: 1000
  - VM2: 1000



A példában most mindegyik virtuális gép egyprocesszoros, egyébként szorozni kéne a processzoraik számával.

Példa a share használatára

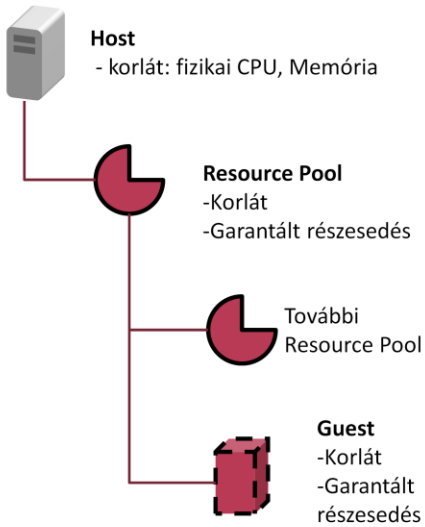
- Több VM-et futtató gép esetén a CPU share értékek a következők
  - VM1: 1000
  - VM2: 1000
  - VM3: 2000



- Versenyhelyzet esetén mennyi a VM1 részére biztosított erőforrás-mennyiség?
  - 1000 / (1000+1000+2000) = 1/4 CPU idő

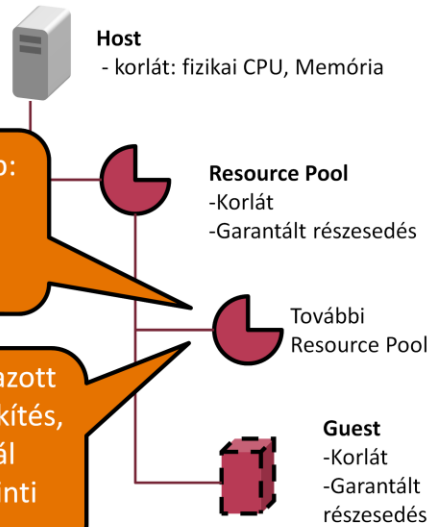
Hierarchikus erőforráskezelés

- Nemcsak virtuális gépek szintjén lehet korlátozni
- Pool-okba szervezhetők a VM-ek
- Használati eset példák:
  - Egy felhasználó összes gépére egy közös korlátozás
  - Egy feladatot ellátó gépek csoportjára korlát
  - Kritikus/nem kritikus alkalmazások csoportosítása



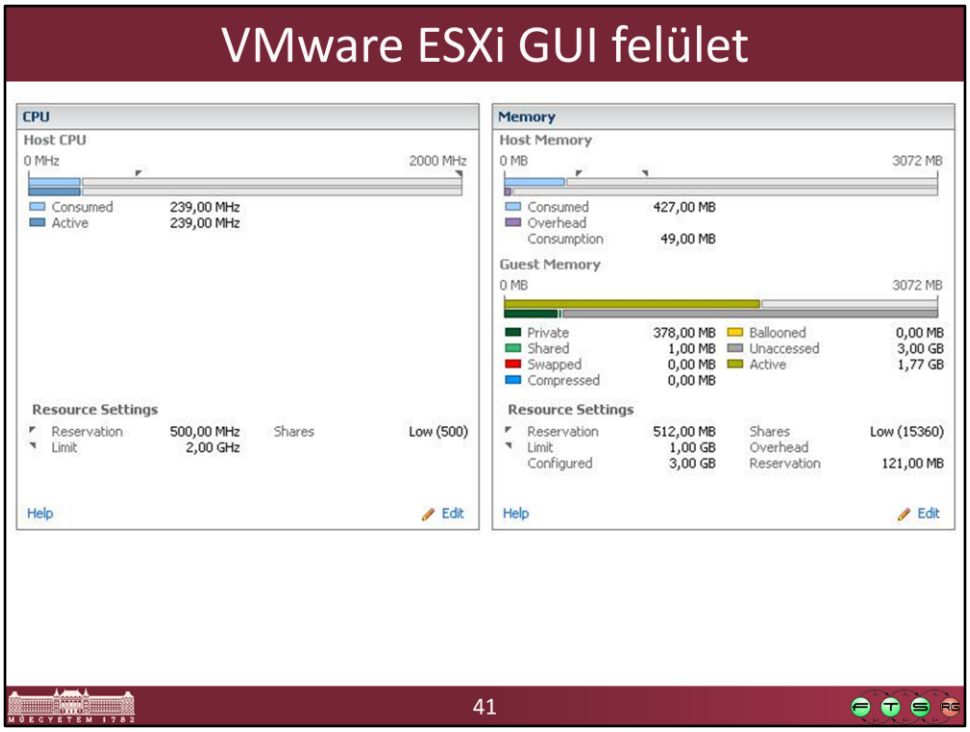
Hierarchikus erőforráskezelés

- Nemcsak virtuális gépek szintjén lehet korlátozni
- Pool-okba szervezhetők a VM-ek
- Használati eset
  - Egy felhasználó gépére egy közö
  - Egy feladatot ellátó gépek csoportjára korl
  - Kritikus/nem kr alkalmazások csoportosítása



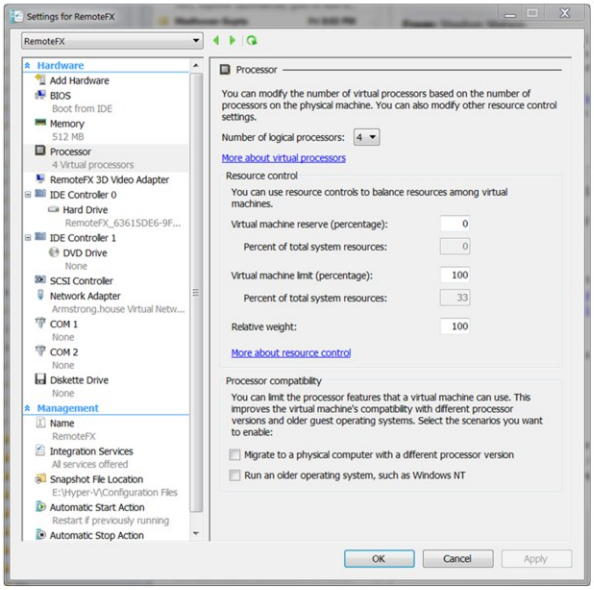
Korlátokat szab:
-Host
-Resource Pool
-Guest

Egymásba ágyazott korlátoknál szűkítés, konfliktusnál prioritás szerinti feloldás



Az ábrán az ESXi felületén látszanak egy konkrét virtuális gépre beállított korlátozások.

Hyper-V: hasonló erőforrás-gazdálkodás



Pl. CPU esetén itt is Reserve / Limit / weight (share) van

DEMO VMware ESXi

- Teljesen távoli elérésen alapul
- „Nagyjából” kompatibilis a Workstation/Player/Server virtuális gépeivel
  (VMware Converter… most nem demózzuk)
- Fejlett erőforrás-gazdálkodás
- Távoli menedzsment PowerShell segítségével
  - Get-VM, Get-Stat…
- Inftech laboron mindenki kipróbálhatja ☺

Tartalom

- Ismétlés (lásd Operációs rendszerek)
  - Virtualizáció fajtái
  - Platform virtualizációs megoldások
  - Kliens oldali virtualizációs igények
- Szerveroldali virtualizáció
  - Platform virtualizációs megoldások
  - Erőforrás-gazdálkodás
  - **Operációs rendszer szintű virtualizáció**

Operációs rendszer szintű virtualizáció

- Kezdetben volt a *chroot*…
  - A fájlrendszer gyökerét átirányíthatjuk egy alkönyvtárra (egy folyamatra vonatkozik!)
  - Ez nem teljes körű izoláció, de sok esetben működik
    - Kernel minden adatszerkezete közös (folyamatlista, hálózati interfész, IP, routing, sysctl beállítások…)
    - A chrootból ráadásul ki is lehet navigálni a VFS adatszerkezeten keresztül…
  - Hogy is néz ki: egy teljes alap OS telepítést készítünk egy alkönyvtárba, ami kicsit eltérő is lehet az eredetitől
  - Problémás könyvtárak: /proc, /sys, /dev, /tmp, /var, …

Operációs rendszer szintű virtualizáció

- Megoldás:
  - Ne látszódjanak ki a kernel singleton erőforrásai…
- Ehhez módosítani kell a kernelt
  - Bevezetni a konténer fogalmát
  - Minden rendszerhívást ellátni a konténer kontextus szerinti válogatással
  - Singleton erőforrásokat dinamikusan példányosíthatóvá alakítani
  - A konténerből kifele mutató referenciák mostantól biztonsági réseknek számítanak!
  - A módosítások ára: 1-2% teljesítményveszteség

Erőforrás-gazdálkodás

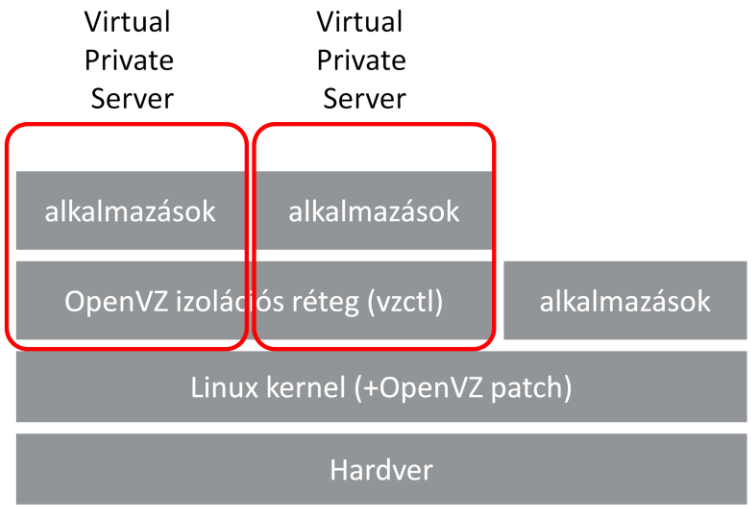
- **CPU** – a kernel beépített ütemezője, prioritáskezelője, kiegészítve szigorú cpuidő-korlátozással
- **Memória** – a kernel beépített memóriakezelője, kiegészítve szigorú méretkorlátozással
- **Háttértár** – a fájlrendszer egy alkönyvtára, quota rendszerrel korlátozható foglalás
- **Hálózat** – a kernel beépített Ethernet hídja vagy routing táblája, pl. IPtables QoS paraméterekkel korlátozható áteresztőképesség
- **Egyéb perifériák** – a kernelben lévő meghajtón keresztül

Operációs rendszer szintű virtualizáció

Tanulság:

+ Az OS szintű virtualizáció nagyon kis költségű,
+ erőforrás virtualizációs és
+ erőforrás gazdálkodási szempontból problémamentes.
- Biztonsági szempontból kevésbé jó izoláció
- Közös kernellel kell élni (azonos verzió, fordítási paraméterek)

OpenVZ architektúrája



OpenVZ képességek

- A VPS belsejében „komplett” telepített OS található
- Egy VPS indításakor a kernel teljesen inicializálatlan állapotban mutatja magát -> saját init scripteket futtat minden VPS
- A VPS-be telepített OS környezet sablonokból (templates) telepíthető le még a VPS indítása előtt
- A VPS-ben lévő fájlok akár meg is oszthatóak több VPS között (hard link!)

A következő rész tartalmából

- Szerver virtualizációs megoldások központi menedzsmentje
  – avagy hogyan építsünk egy teljes infrastruktúrát virtuális gépekre
- Finom funkciók
  - Live migration
  - Hibatűrés
  - Terheléselosztás
  - Sablonkezelés
  - …és a már megszokottak: monitorozás, hozzáférés-kezelés…

Összefoglalás

- Virtualizáció alap funkció lett
  - Kliens és kiszolgáló oldalon is
- Fejlett megoldások
  - Hypervisor egyre inkább alap komponens
- További információ:
  - Virtualizációs technológiák és alkalmazásaik választható tárgy (VIMIAV89)

- VIMIAV89: https://www.inf.mit.bme.hu/edu/courses/virttech