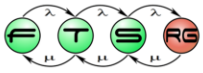
Virtualizációs technológiák és alkalmazásaik

Menedzsment feladatok virtualizált környezetben

Tóth Dániel, Micskei Zoltán



Budapesti Műszaki és Gazdaságtudományi Egyetem
Méréstechnika és Információs Rendszerek Tanszék

Utolsó módosítás: 2010. 11. 25.

Tartalom

- **Virtuális gépek életciklusa**
  - Sablonok
  - Appliance csomagok
  - Automatikus életciklus kezelés
- Erőforrás gazdálkodás
  - Terheléselosztás hosztok között
  - Energiatakarékosság
- Hibatűrő működés
  - Különféle hibamódok
  - Védekezési lehetőségek a meghibásodások ellen

Létrehozás: előállít egy virtuális gép példányt, lefoglalja a megfelelő erőforrásokat, felveszi a nyilvántartásba
Üzembeállítás: a felhasználó számára átadható használható állapotba helyezi: OS telepítve és konfigurálva, hálózat beállítva, távoli hozzáférés, felhasználói fiók/jelszó stb.
Üzemen kívül helyezés: átmenetileg nincs szükség rá, leállítás, de nyilvántartásban marad, gyorsan újraindítható
Visszavonás: virtuális gép nyilvántartásból kivétele, háttértár adatok törlése vagy archiválása

Virtuális gépek üzembeállítása

- Motivációs példa

Virtuális gépek üzembeállítása

- Megoldás:
  - Készítsünk alap virtuális gépeket alap OS telepítéssel és azt másoljuk le, amikor kell
  - Mi ezzel a baj?
    - Testreszabás (IP cím, hosztnév, UUID, SID stb.)
    - Licenszkérdések
    - Túl sok manuális lépés
  - Vezessük be a „**sablon**” (template) fogalmát
    - Olyan, mint egy sima virtuális gép, csak fel van készítve rá, hogy automatikusan üzembeállítható legyen
    - Az üzembeállításhoz konfigurálni kell a vendég OS-t. Mi kell ehhez?
      - Operációs rendszer specifikus ágens (pl.: VMware Tools)

Több megoldás is lehetséges: OS szintű virtualizációnál pl a virtuális gép létrehozása már egyben az OS fájlrendszer példány előállításával is jár, tehát nincs „üres gép” állapot. Ilyenkor a konfiguráció a fájlrendszerben elvégezhető az első indítás előtt, nem kell külön ágens.

Virtuális gépek automatikus üzembeállítása

- Miért álljunk meg az operációs rendszer szintjén?
  - Lehet kész sablonunk a telepített alkalmazásokkal is
  - Az automatikus konfigurálása (még) nem teljesen megoldott
- Nekünk kell a sablonokat elkészíteni?
  - Nagyvállalati környezetben belefér
  - Elérhetőek *Virtual Appliance*-ek, készre telepített gépek, egy specifikus alkalmazás ellátására
  - Vannak csoportos „Appliance Team”-ek is
    - Pl.: 3 rétegű webes alkalmazásszerver 3 VM-ből egy csomagban készre telepítve
    - VMware vApp (bővebben: http://blogs.vmware.com/vapp/)
    - VMware Studio alkalmazással készíthetők

„Újhullámos” infrastruktúramenedzsment

- Egy virtuális gép mostantól kezdve egy építőelem
  - (FRU - Field Replacable Unit)
  - Szükség esetén példányosítható sablonból
  - Feladata végeztével eldobható
- Virtual appliance-ekből összeépíthető a teljes infrastruktúra
  - Anélkül, hogy alkalmazás telepítéssel, konfigurálással bajlódni kéne
  - Konfigurációmenedzsment problémáját is meg lehet oldani ezen a szinten
- Ez az egész MOST kezdődik az iparban!

Példa: VMware LabManager

- Automatikus életciklus kezelés - Miért jó ez?
  - Felhasználó is elvégezheti saját magának
  - Szabályokkal korlátozható a felhasználók VM használata (pl. lejárati idő, nem használt VM-ek leállítása stb.)
- Appliance-ek használata
  - Pl.: a LabManager a virtuális hálózatok közötti átjárást egy-egy kis Linux alapú NAT appliance-szel oldja meg
- A LabManager csak ESX szervereket tud felügyelni
- **UPDATE**: „private cloud” megoldás → lásd később

A lab management alkalmazások és elnevezés 2-3 éve volt divatos, most „private cloud”-nak hívják az ilyesmi (vagy ehhez alapjaiban nagyon hasonló) megoldásokat. Ott annyival egészül ki, hogy tényleg mindent automatizálunk, és lehetőség van nyilvános cloud szolgáltatásokhoz való csatlakozásra.

Tartalom

- Virtuális gépek életciklusa
  - Sablonok
  - Appliance csomagok
  - Automatikus életciklus kezelés
- **Erőforrás gazdálkodás**
  - Terheléselosztás hosztok között
  - Energiatakarékosság
- Hibatűrő működés
  - Különféle hibamódok
  - Védekezési lehetőségek a meghibásodások ellen

Erőforrás gazdálkodás

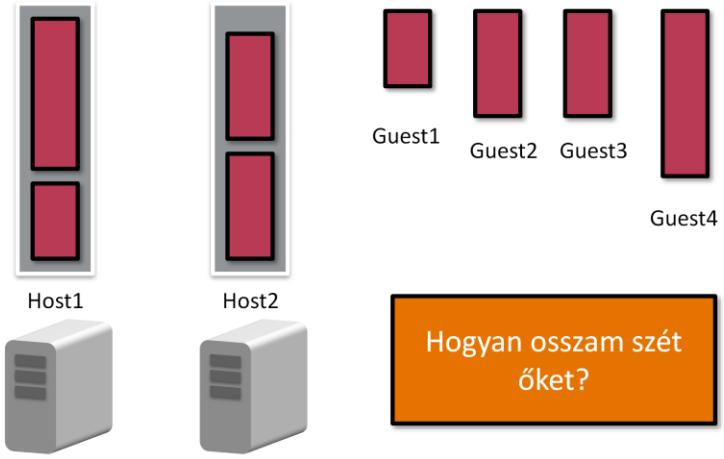
- Tipikus probléma: az egyik nagy magyarországi bankban…
  - 80db ESX hoszt
  - 400 - 1000db közötti virtuális gép
  - Két fő telephely
  - Egy üzemeltetési rémálom…
  - … lenne megfelelő központi menedzsment nélkül

- Agilitás
- Konszolidáció
- Közelítőleg megvan a 10:1 arány

Gondoljunk rá, hogy egy ekkora rendszerben garantáltan folyamatosan van valami meghibásodás!
Az adatok nem légből kapottak, az egyik 2008-as VMware Users Group meetingen hangzottak el.
Agilitás – gyorsan képes követni a pillanatnyi igényeket

Erőforrás gazdálkodás

- Allokációs probléma (pl. memória foglalás szerint)



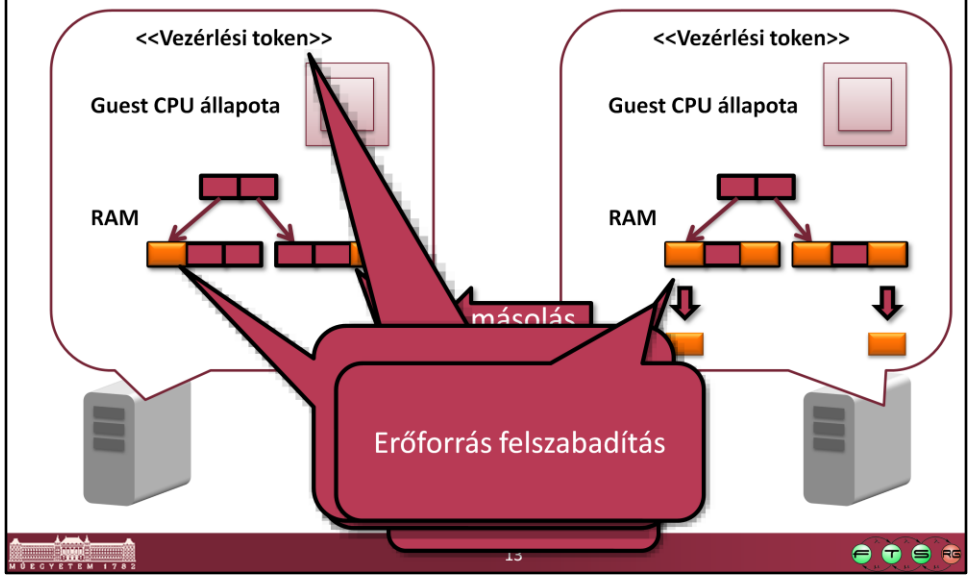
Erőforrás gazdálkodás

- Manuálisan nehéz feladat
  - Főleg sok hoszt és sok guest esetén problémás
  - Menet közben is változhat az erőforrás foglalás (főleg CPU, de memória esetén is)
  - Többféle optimalizálási cél is lehet
    - Hosztok egyenletes terhelése (guestek teljesítményét maximalizálja)
    - Minimális számú hoszt használata (energiatakarékosság)
- VMware DRS (Distributed Resource Scheduling)
  - Fürtökbe fog sok ESX/ESXi hosztot
  - Automatikusan vagy félautomatikusan osztja szét a guesteket a hosztok között
  - Menet közben a változó terhelésekre állítható gyorsasággal reagálva is változtathatja a hozzárendelést
    - hogyan lehetséges ez?

DRS félautomatikus üzemmód: javaslatot tesz, amit manuálisan lehet elfogadni vagy felülbírálni

Működő virtuális gépek áthelyezése

- **Live migration** - Hogy is működik?



Tartalom

- Virtuális gépek életciklusa
  - Sablonok
  - Appliance csomagok
  - Automatikus életciklus kezelés
- Erőforrás gazdálkodás
  - Terheléselosztás hosztok között
  - Energiatakarékosság
- **Hibatűrő működés**
  - Különféle hibamódok
  - Védekezési lehetőségek a meghibásodások ellen

Hibatűrés

- Hibatűrés célja:
  - Szolgáltatás nyújtása meghibásodás esetén
  - Komplex feladat
- Első lépés:
  - Hibatípusok azonosítása
  - Mindegyikhez megfelelő védekezés kitalálása

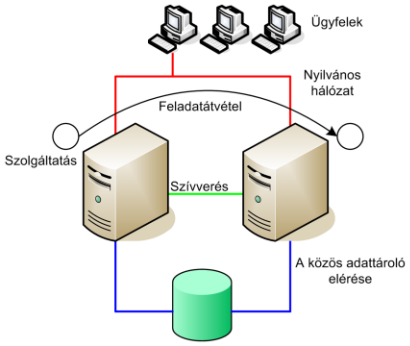
Szolgáltatásbiztonságra tervezés - https://www.vik.bme.hu/kepzes/targyak/VIMIM146/

Példák szolgáltatás kiesésekre

	Nem tervezett	Tervezett
Környezet / emberek	-Hibás üzemeltetői tevékenység -Támadás -Elemi kár	
Alkalmazás	-Alkalmazás leáll -Adatok inkonzisztenssé válnak	- Alkalmazás verzióváltás
OS	-OS hiba	- OS frissítés miatt újraindítás kell
HW	-HW alkatrész meghibásodik -Hálózat kiesés -Tápellátás megszűnik	-HW-t karban kell tartani

HW hiba kezelése – klasszikus eset

- Hiba elfedése
  - Redundancia (2. táp, RAID, több hálózati út…)
- Ha nem sikerül gép szinten elfedni
  - Pl.: feladatátvételi fürtök
    - Szolgáltatás átvétele
    - Tervezett leállásra is jó
    - Rövid kiesés van



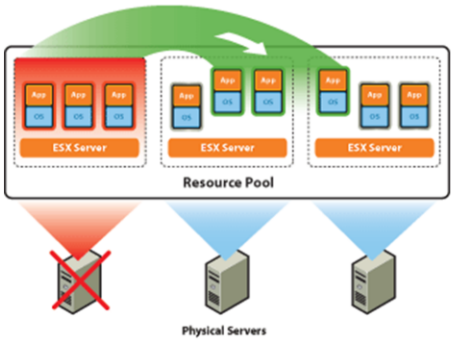
- …

HW hibák kezelése – virtualizáció

- Problémák virtualizáció esetén:
  - A hoston futó összes guest memória és CPU állapotát elveszítjük -> guest leállási hiba
  - Egy HW hiba esetén **SOK** virtuális gép hibásodik meg
  - Live migráció „azellen nemvéd”, csak a **tervezett leállások előtt** lehet leköltöztetni a guesteket egy hosztról

HW hibák kezelése – virtualizáció

- Ha a guest háttértára hozzáférhető marad, akkor újraindíthatjuk másik hoszton (pl. VMware HA)



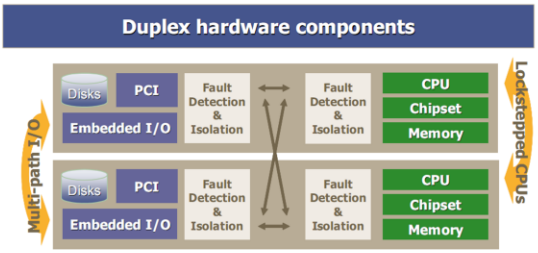
- Tulajdonképpen egy speciális feladatátvételi fürt
- „Host clustering” (vö. guest clustering)

• Ha a guest OS és alkalmazások fel voltak készítve erre („crash konzisztencia”), akkor újraindítás után folytathatják a végrehajtást
• A leállást közvetlenül megelőző utolsó állapot nem biztos, hogy reprodukálható, de ez nem is mindig fontos
• Ha a vendég OS vagy alkalmazások szintjén volt hibatűrő fürtözés, akkor ez ennek egy kiegészítő megoldása lehet (ne fogyjanak el a fürt tagjai)

Kép forrása: http://www.vmware.com/products/server/landing.html

HW hibák kezelése – klasszikus eset 2.

- Futási állapot elvesztés kivédése
  - Checkpointing
    - rendszeresen állapotmentést készítünk, leállás után a legutóbbi ép állapotmentést visszatöltjük
    - Alkalmazás szintű megoldás!
    - Pl. SA Forum Checkpoint API
  - Lockstep (pl. Stratus ftServer)



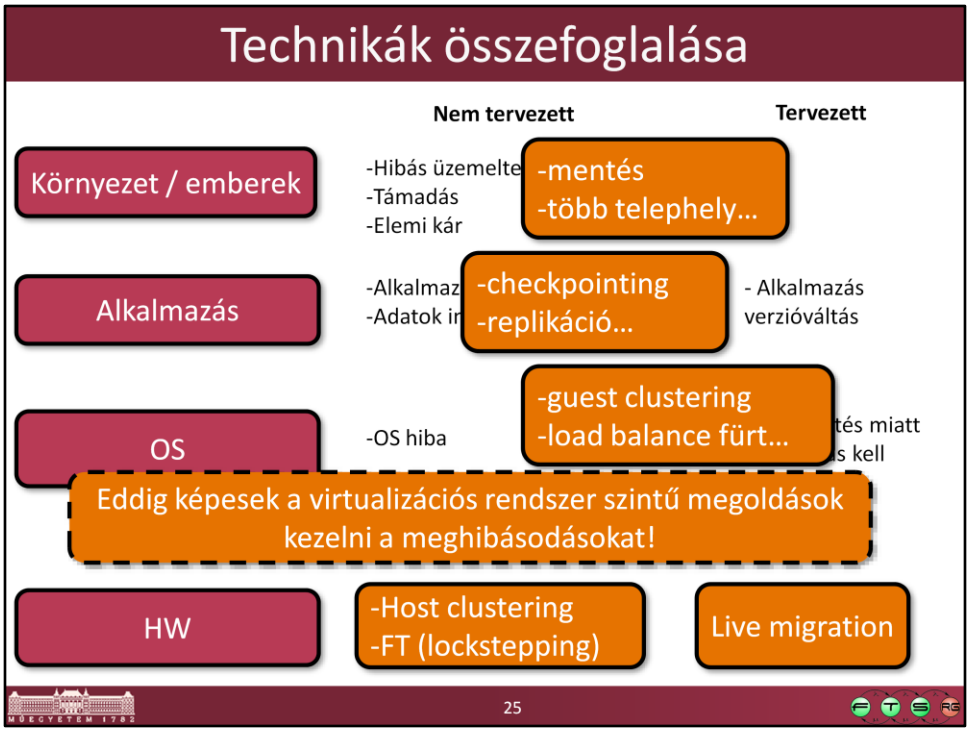
HW hibák kezelése – virtualizáció 2.

- Többszörözött futtatás több hoszton (lockstep)
  - Azonos guest gép több példánya több hoszton.
    Több példány = azonos memóriatartalom és CPU állapot!
  - Egy példány „elsődleges”, ez kommunikál a hálózaton
  - A többi példány „tartalék”, ezek kívülről nem megfigyelhető módon (kis késletetéssel) követik az első állapotát
  - Előny: külső megfigyelők nem veszik észre a váltást
  - Hátrány: nagyon költséges, teljesítményvesztés, több példány
  - Nem véd: guest gép szoftverhibája ellen – minden példány egyformán bele fog futni ugyanabba a hibába

Többszörözött futtatás

- Megvalósítás (VMware FT, Xen Remus)
  - Feltételezzük, hogy minden példány CPU-ja egyformán determinisztikusan működik
    - Több virtuális CPU között már versenyhelyzet lehet – csak 1 vCPU lehet!
  - Egyszer a futás során történik egy teljes szinkronizáció
  - Rögzíteni kell minden külső eseményt, ami az elsődleges példánnyal történik
    - Megszakítások a virtuális perifériáktól
    - Hálózati csomagok érkezése
  - Rögzíteni kell az események bekövetkeztekor a CPU állapotát (pontosan melyik utasításon állt)
    - megtehető, az események érkezésekor a VMM eleve állapotmentést csinál
  - Vissza kell játszani az eseményeket a tartalék példányon pontosan a megfelelő utasításhelyre elhelyezett trapekkel
    - Csak bináris fordítással valósítható meg
  - A tartalék valamennyit késik az elsődlegeshez képest
    - Addig vissza kell tartani az elsődleges példány kimenő hálózati forgalmát, amíg a tartalék nem jutott el a küldés állapotig (miért is? – „árva állapot”)

További információ: Xen Remus - http://dsg.cs.ubc.ca/remus/



A fentieken kívül természetesen még rengeteg hibatűrést, rendelkezésre állást garantáló technika van.