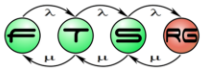
Intelligens rendszerfelügyelet

Virtualizáció – Központi menedzsment

Micskei Zoltán, Tóth Dániel



Budapesti Műszaki és Gazdaságtudományi Egyetem
Méréstechnika és Információs Rendszerek Tanszék

Utolsó módosítás: 2011. 04. 28.

Tartalom

- **Központi menedzsment – alap infrastruktúra**
  - Menedzsment szerver
  - Hozzáférés-kezelés
  - Közös hálózat, tárhely
- Erőforrás-gazdálkodás
  - Allokációs problémák
  - Terheléselosztás fizikai gépek között
- Hibatűrés
  - Különféle hibamódok
  - Védekezési lehetőségek a meghibásodások ellen
- Virtuális gépek életciklusa
  - Sablonok
  - Automatikus életciklus kezelés

Központi menedzsment motivációs példa

- Ipari esettanulmány banki környezetből
  - 80db ESX gép
  - 400 - 1000db közötti virtuális gép
  - Két fő telephely
  - Egy üzemeltetési rémálom…
  - … lenne megfelelő központi menedzsment nélkül

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

3

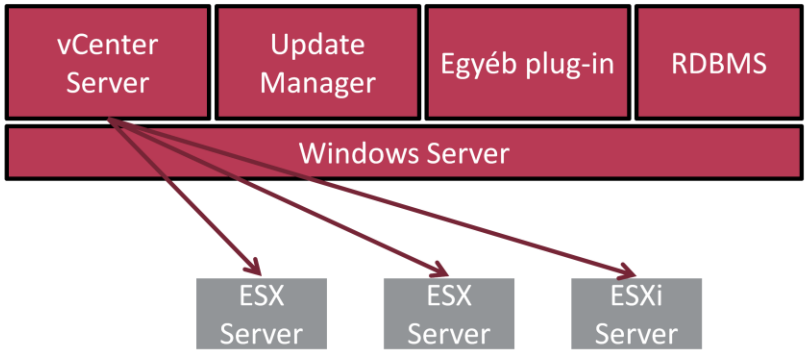
Gondoljunk rá, hogy egy ekkora rendszerben garantáltan folyamatosan van valami meghibásodás!
Az adatok nem légből kapottak, az egyik 2008-as VMware Users Group meetingen hangzottak el.
Agilitás – gyorsan képes követni a pillanatnyi igényeket

(Központi) menedzsment szerver

- Virtualizációt nyújtó gépek összefogása
  - Akár több gyártó megoldását is
- Közös leltár és térkép
  - Fizikai/virtuális gépek, hálózat, felhasználók…
  - Historikus adatok gyűjtése is
- Plusz funkciók
- Pl.: VMware vCenter, MS SCVMM, XenServer…

DEMO vCenter

- A VMware vCenter lesz a futó példa, mostantól kezdve ezen mutatunk be mindent



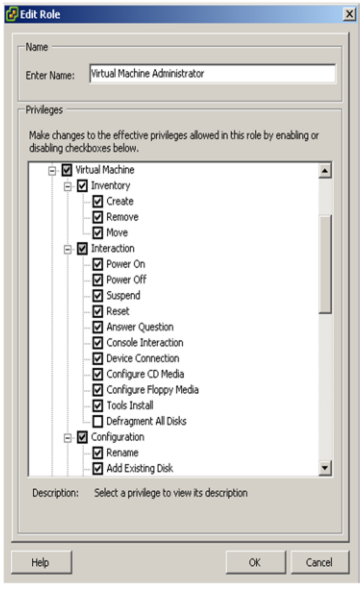
Távoli elérés – Protokoll

- Vezérlés:
  - saját WebService alapú távoli API
    (van hozzá WSDL is a VI SDK-ban, ~1500 osztályból áll)
  - HTTPS felett
    - biztosít: hitelesítés, bizalmas és sértetlen csatornát
  - Az ESXi és a vCenter is ugyanazt a protokollt használja
  - Ezen kívül újabban van WS-Mananagement
    (DMTF SMASH ajánlás alapján)
- Konzol hozzáférés:
  - MKS protokoll
  - Ez valójában egy saját wrapperbe becsomagolt VNC
  - A wrapper biztosít hitelesítést, bizalmas és sértetlen csatornát

SMASH: System Management Architecture for Server Hardware,
http://www.dmtf.org/standards/mgmt/smash/

Felhasználó- és jogosultságkezelés

- **Felhasználókezelés**
  - Active Directory
  - Ez azt jelenti, hogy csoportok is vannak, amiknek lehetnek csoportok tagjai (RBAC megvalósítható)
- **Jogosultsági modell**
  - Hierarchikus fa szerkezetbe szervezett erőforrások (VM, Resource Pool…)
  - Örökölhető engedélyek
  - Hozzáférési maszk 142-féle műveletet definiál (v3.5)
    - Hoszt konfiguráció, VM konfiguráció, adattárak, stb.

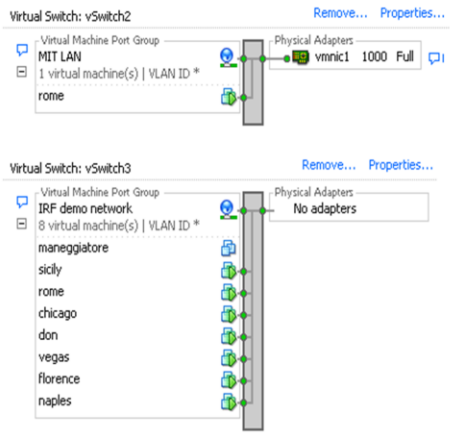


Szervereknél fontos a hozzáférés kezelés

- A virtuális szerver konzol távoli elérése = „fizikai” hozzáférés a virtuális géphez
- Két fontos részfeladat:
  - Felhasználókezelés
  - Engedélyezés
- A virtuális gépekhez felhasználók rendelhetőek
  - Megadható, hogy milyen műveletet végezhetnek…
  - Milyen műveletek vannak?
  - Sok gép esetén valamilyen módon kezelhetővé kell tenni…

Virtuális hálózat

- Virtuális switch-ek, elnevezett hálózatok
- Csak bridge-elt és host-only módot támogat
  - Ha NAT kell, akkor azt saját VM-ben kell megoldanunk
- Virtuális switch-hez rendelhetőek a VM-ek és a Service Console, VMkernel hálózati interfészei is
- Virtuális switch-hez fizikai hálózati kapcsolat rendelhető
  - Akár redundánsan is



Közös tárhely

- Az adatokat lokális diszk helyett SAN-on tároljuk
- Többszörös hozzáférési lehetőség
- Dinamikus allokáció
- Alacsonyabb fajlagos költségek
- Egy bizonyos méret fölött virtualizáció nélkül is megjelenik ez a megoldás (aka. *„Tárhely virtualizáció”*)

- Tipikus protokollok: FC, iSCSI, NFS…

Központi menedzsment

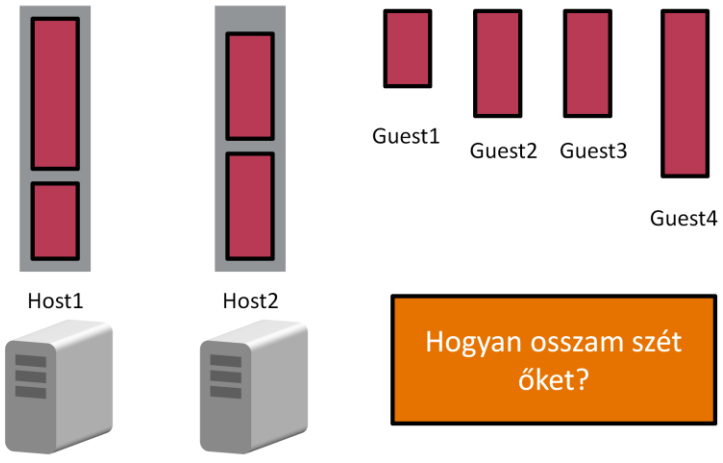
- Ha csak ennyit tudna, azzal még sokat nem érnénk…
- Új szolgáltatások
  - Gépek fürtbe szervezése (Cluster)
  - Virtuális gépek áthelyezése gépek között
  - …akár működés közben (*live migration*)
  - Hibatűrés
  - Terheléselosztás
  - …

Tartalom

- Központi menedzsment – alap infrastruktúra
  - Menedzsment szerver
  - Hozzáférés-kezelés
  - Közös hálózat, tárhely
- **Erőforrás-gazdálkodás**
  - Allokációs problémák
  - Terheléselosztás fizikai gépek között
- Hibatűrés
  - Különféle hibamódok
  - Védekezési lehetőségek a meghibásodások ellen
- Virtuális gépek életciklusa
  - Sablonok
  - Automatikus életciklus kezelés

Erőforrás gazdálkodás

- Allokációs probléma (pl. memória foglalás szerint)



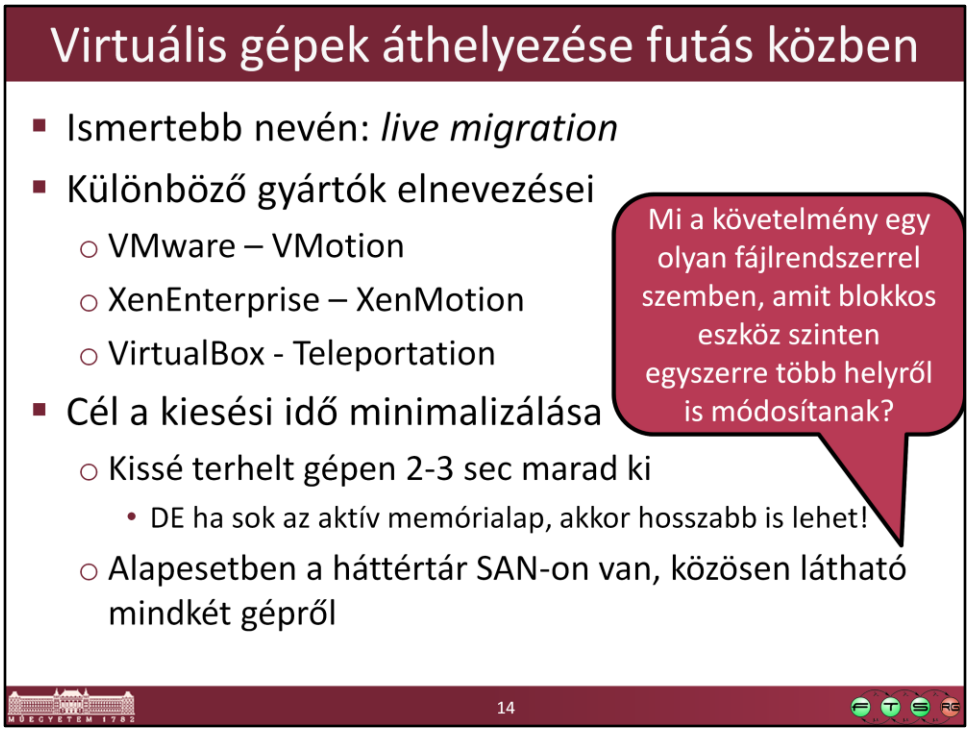
Nagyon szép lineáris programozási feladatokra vezethető vissza…

Erőforrás gazdálkodás

- Manuálisan nehéz feladat
  - Főleg sok fizikai és virtuális gép esetén problémás
  - Menet közben is változhat az erőforrás foglalás (főleg CPU, de memória esetén is)
  - Többféle optimalizálási cél is lehet
    - Hosztok egyenletes terhelése (VM teljesítményét maximalizálni)
    - Minimális számú hoszt használata (energiatakarékosság)
- VMware DRS (Distributed Resource Scheduling)
  - Fürtökbe fog sok ESX/ESXi gépet
  - Automatikusan vagy félautomatikusan osztja szét a VM-eket a fizikai gépek 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

Ez sem „csodaszer”:
- Egy virtuális gépet nem fog tudni szétszórni egynél több hosztra
- Nem helyettesíti az alkalmazás szintű terheléselosztó rendszereket
- Magas szintű QoS metrikákra nem tud szabályozni

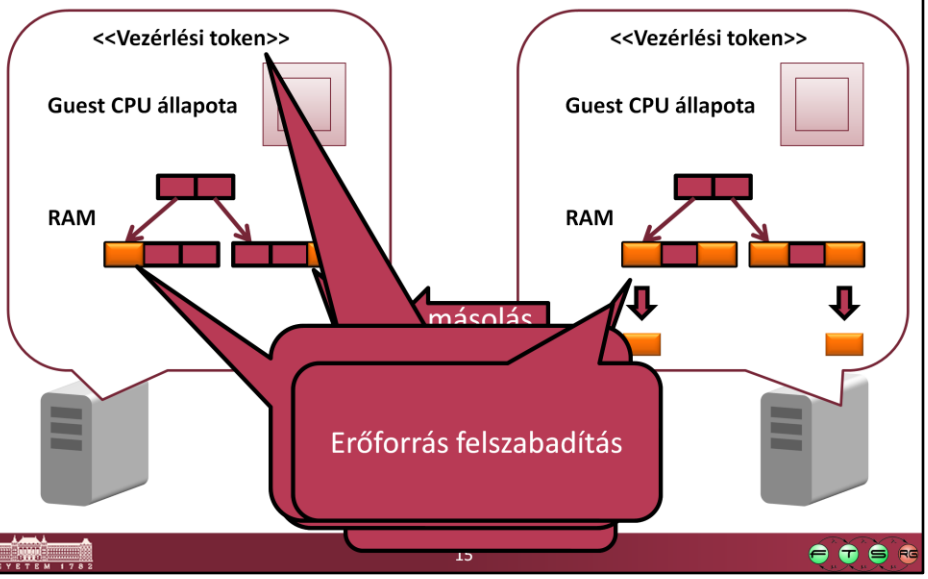


Kölcsönös kizárást és tranzakció-kezelést kell biztosítania a fájlrendszernek a blokkos adatformátum szintjén, hogy a metaadatok módosítása során a különböző helyeken átmenetileg se láthassanak inkonzisztens állapotot.

Létezik háttértárat mozgató megoldás is (Storage VMotion), működési elve megegyezik a memóriamozgatáséval.

Virtuális gépek áthelyezése

- Hogy is működik?



Tartalom

- Központi menedzsment – alap infrastruktúra
  - Menedzsment szerver
  - Hozzáférés-kezelés
  - Közös hálózat, tárhely
- Erőforrás-gazdálkodás
  - Allokációs problémák
  - Terheléselosztás fizikai gépek között
- **Hibatűrés**
  - Különféle hibamódok
  - Védekezési lehetőségek a meghibásodások ellen
- Virtuális gépek életciklusa
  - Sablonok
  - Automatikus életciklus kezelés

Hibatűrés

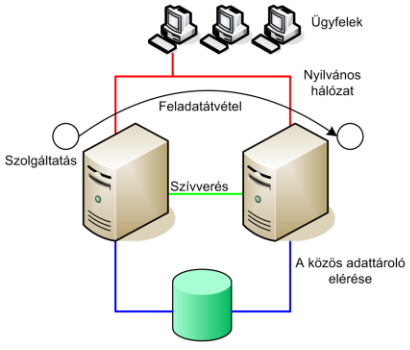
- Hibatűrés célja:
  - Szolgáltatás nyújtása meghibásodás esetén
  - Komplex feladat
- Első lépés:
  - Hibatípusok azonosítása
  - Mindegyikhez megfelelő védekezés kitalálása

Példák szolgáltatás-kiesésekre

	Nem tervezett	Tervezett
Környezet / emberek	-Hibás üzemeltetői tevékenység -Támadás -Elemi kár	
Alkalmazás	-Alkalmazás leáll -Adatok inkonzisztenssé válnak	- Alkalmazás verzióváltás
OS	-OS crash	- 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



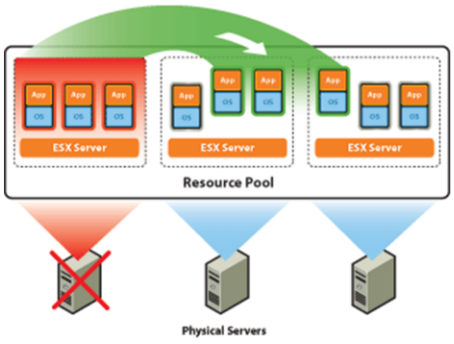
- …

HW hibák kezelése – virtualizáció

- Problémák virtualizáció esetén:
  - A fizikai gépen futó összes VM memória és CPU állapotát elveszítjük -> VM leállási hiba
  - Egy HW hiba esetén **SOK** virtuális gép hibásodik meg
  - Live migration „azellen nemvéd”, csak a **tervezett leállások előtt** lehet leköltöztetni a VM-eket egy fizikai gépről

HW hibák kezelése – virtualizáció

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



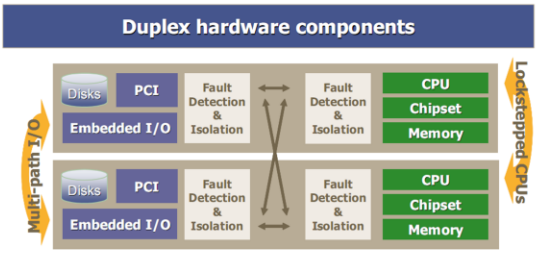
- Tulajdonképpen egy speciális feladatátvételi fürt
- „Host clustering” (vö. guest clustering)

• 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

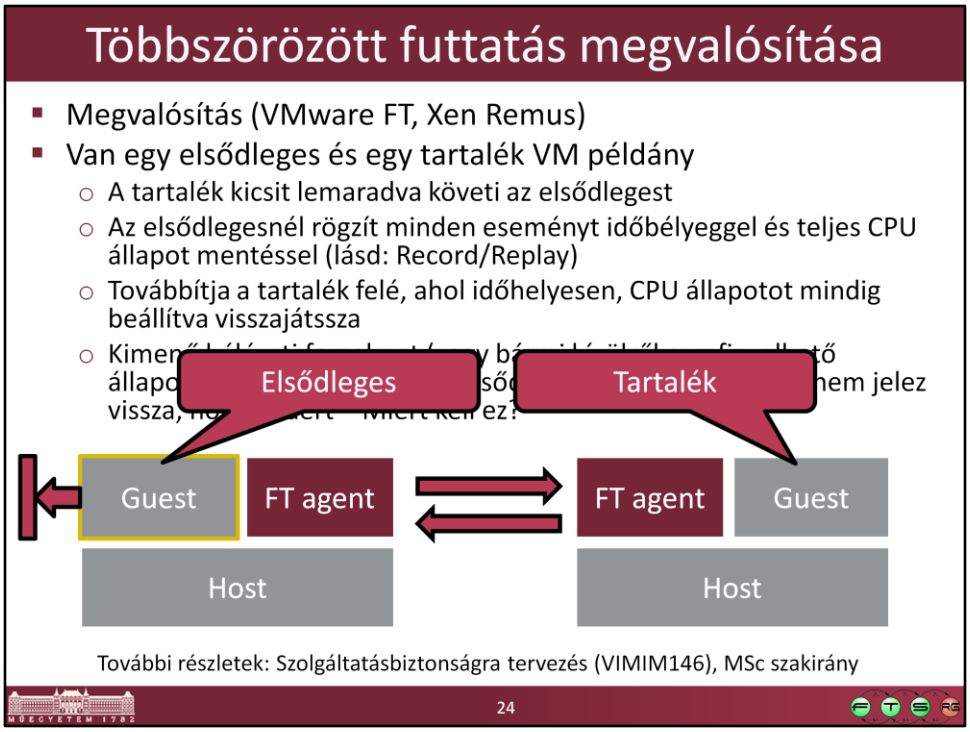
HW hibák kezelése – klasszikus eset 2.

- 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 VM 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: teljesítményvesztés, költséges (több példány)
  - Nem véd: VM szoftverhibája ellen – minden példány egyformán bele fog futni ugyanabba a hibába



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

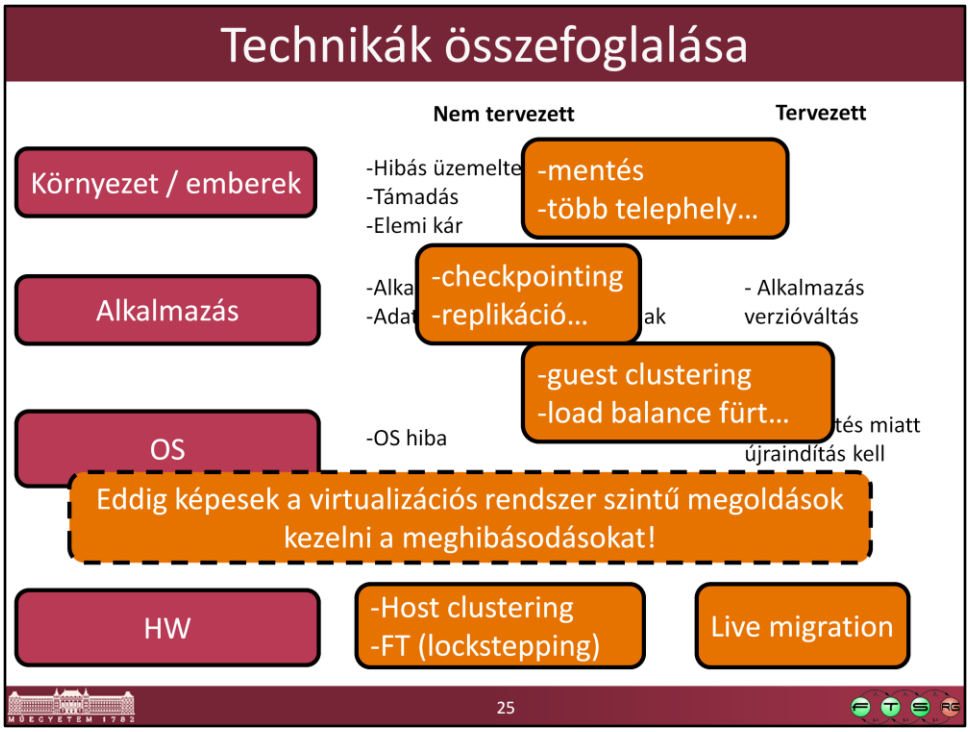
Vissza kell játszani az eseményeket a tartalék példányon pontosan a megfelelő utasításhelyre elhelyezett trapekkel

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/



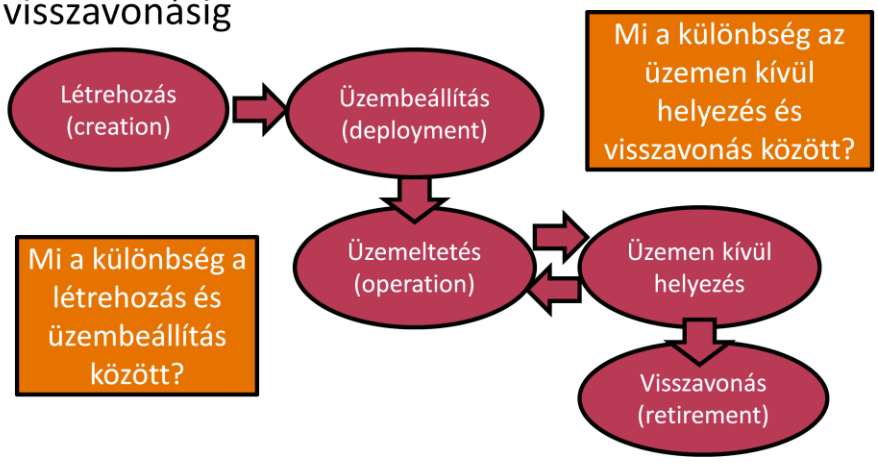
A fentieken kívül természetesen még rengeteg hibatűrést, rendelkezésre állást garantáló technika van.

Tartalom

- Központi menedzsment – alap infrastruktúra
  - Menedzsment szerver
  - Hozzáférés-kezelés
  - Közös hálózat, tárhely
- Erőforrás-gazdálkodás
  - Allokációs problémák
  - Terheléselosztás fizikai gépek között
- Hibatűrés
  - Különféle hibamódok
  - Védekezési lehetőségek a meghibásodások ellen
- **Virtuális gépek életciklusa**
  - Sablonok
  - Automatikus életciklus kezelés

Virtuális gépek életciklusa

- Életciklus - a virtuális gép létének állapotai a létrehozástól az üzemeltetésen keresztül a visszavonásig



- 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, gépnév, UUID, SID stb.)
    - Licensz kérdések
    - 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 igazán 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
- **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.

Összefoglalás

- Virtualizáció fogalma, alapvető megvalósításai
- A platform virtualizáció desktopon és szerveren
- Az operációs rendszer szintű virtualizáció
- A szerver virtualizáció különleges követelményei
- A szerver virtualizáció központi menedzsmentje
- A szerver virtualizáció szolgáltatásainak kiterjesztése, hibatűrés, terheléselosztás
- Virtuális gép sablonok és appliance-ek