Virtualizációs technológiák
Mérési útmutató

Informatikai technológiák laboratórium I.
összeállította: Tóth Dániel
Méréstechnika és Információs Rendszerek Tanszék

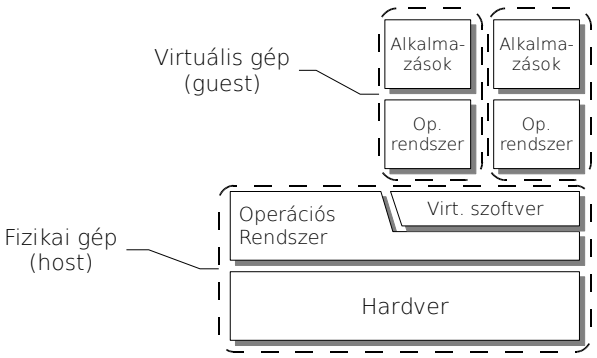
2012. február 28.

A mérés célja

A mérés célja, hogy a hallgatók megismerkedjenek egy nagyvállalati szervervirtualizációs megoldással, annak használatával és alapvető karbantartásával. A mérés során VMware VSphere virtualizációs platformon kell feladatokat végezni. A feladatok részben a virtualizációs rendszer konfigurálásához, részben pedig a virtuális gépek létrehozásához, módosításához kapcsolódnak, egy rövid reprezentatív áttekintést adva egy nagyvállalati virtualizációs rendszer mindennapi használatáról. A mérés során néhány egyszerű teljesítménymérésre is sor kerül, aminek célja, hogy egy alapszintű elképzelés alakuljon ki a virtualizációs rendszerek teljesítményéről, illetve az erőforrásgazdálkodási lehetőségek teljesítményre gyakorolt hatásáról. A mérés időtartama nyilvánvalóan nem elegendő teljeskörű, statisztikailag is elfogadható, következtetések levonására alkalmas teljesítménymérések kivitelezéséhez, inkább csak példákat mutat be arra, hogy milyen lehetőségek vannak egy ilyen mérés kivitelezésére.

1. Bevezető

Virtualizációnak nevezzük az olyan technológiákat, melyek lehetővé teszik, hogy egy fizikai számítógép vagy annak valamilyen hardver erőforrása több, esetleg eltérő típusú virtuális számítógépnek illetve erőforrásnak látszódjék a gépen futó szoftverek számára.



1. ábra. Platform virtualizáció felépítése

Az első virtualizációs megoldásokat az IBM fejlesztette ki a 60-as években a CP-40-re épülő nagygépes rendszereiben. A modern operációs rendszerek is egyfajta *erőforrásvirtualizáció*t valósítanak meg azzal, hogy elfedik a hardvert a gépen futó folyamatok elől, és egy egységes szoftveres interfészt biztosítanak az erőforrások dinamikus lefoglalására és elérésére. Teljes számítógép

virtualizálása (ún.: *platformvirtualizáció*) oly módon, hogy a gépen több operációs rendszer is futhasson, sokáig csak nagygépes (mainframe) környezetben volt elérhető. Ennek egyik fő oka, hogy a virtualizáció különleges követelményeket[1] támaszt a processzor utasításkészletével, a memória kezelő egységgel (MMU, Memory Management Unit) illetve a perifériák programozói felületével szemben. A hagyományos PC-kben található x86-os processzor architektúra tervezésekor ezeket a követelményeket nem vették figyelembe, ezért PC-re platformvirtualizáció csak nagyjából a '90-es években, különleges szoftveres megoldásokkal (futás közbeni kód módosítás, *binary translation*) vált elérhetővé. Ilyen megoldást sokáig a piacon egyedül csak a VMware cég szállított. A virtualizációs megoldások rohamosan bővülő piacán időközben számos más szereplő is megjelent (Xen, Parallels, Oracle, Microsoft). 2005 környékén megjelentek az x86 utasításkészlet olyan kiegészítései (Intel VT-x, AMD-V), melyekkel már a futtatott virtuális gép kódjának átalakítása nélkül is lehetőség nyílik a platformvirtualizációra. Hasonló fejlesztések zajlanak más CPU architektúrák esetén is.

A virtualizációtól némileg megkülönböztetendő ágat képviselnek az *alkalmazás szintű futtatókörnyezetek*, amelyek – futási időben történő teljes kód újrafordítással – szoftverek futtatását teszik lehetővé eltérő architektúrájú processzorokon. Ennek legismertebb példái a JavaVM (Java Virtual Machine) és a Microsoft .NET CLR (Common Language Runtime). A fentebb ismertetett platformvirtualizáció legnagyobbrészt a hardverrel azonos architektúrájú processzort biztosít a virtuális gépek számára, a JavaVM és a .NET CLR azonban a hardvertől teljesen eltérő utasításkészletet biztosít az alkalmazások számára.

Általában véve elmondható, hogy a processzor *emulációval* működő virtuális gép futtatás és a *virtualizáció* között a fő különbség, hogy az utóbbi esetben a vendég gépen kiadott utasítások túlnyomó többsége változtatás nélkül fut a gazdagép processzorán.

Léteznek még a virtualizációnak egyéb fajtái is, melyek valamilyen korlátozott formáját valósítják meg, és leginkább az *erőforrás-virtualizáció* körébe sorolhatóak be. Legfontosabb az *operációs rendszer szintű virtualizáció*, másnéven *konténer-alapú virtualizáció*, melynek lényege, hogy egyazon operációs rendszeren belül elkülönített végrehajtási környezeteket (*container*, *jail*) alakítunk ki, amelyek nem tudnak egymás létezéséről, elkülönített erőforráskészlettel (pl. fájlrendszer, hálózati portok) rendelkeznek, úgy viselkednek, mintha külön operációs rendszereken futnának, ám a kernel valójában közös. Ilyen megoldások például az OpenVZ, Linux VServer, Solaris Containers, AIX Workload Partitions. Az operációs rendszer szintű virtualizációt leginkább az különbözteti meg a platform virtualizációtól, hogy az előbbi az operációs rendszer erőforrásainak virtualizálását jelenti, míg az utóbbi hardver erőforrásokét. Természetesen léteznek a kettőt kombináló megoldások is.

*Alkalmazásvirtualizáció*nak[2] nevezik az olyan operációs rendszer szintű virtualizációt megvalósító termékeket, amelyek kifejezetten azt a célt szolgálják, hogy nagy és komplex alkalmazásokat telepítés nélkül lehessen futtatni egy operációs rendszer felett. Az izolált környezetek közösen használják az operációs rendszer kerneljét, ám elkülönített programkönyvtárakat és konfigurációs fájlokat (Windows esetén Registry-t) kapnak, ami egy önhordó, egyszerűen elindítható csomagban terjeszthető. Ilyen termékek például a VMware ThinApp, vagy a Microsoft App-V.

2. A virtualizáció szerepe az IT-ben

A virtualizáció nagyvállalati környezetben megfigyelhető nagyszabású terjedéséért elsősorban három fő szempont felelős:

- Erőforrás-konszolidáció
- Szeparáció (hibatűrés és biztonság)

[1] Gerald J. Popek and Robert P. Goldberg (1974). "Formal Requirements for Virtualizable Third Generation Architectures". Communications of the ACM 17 (7): 412 -421

[2] Nem összetévesztendő az alkalmazás szintű futtatókörnyezetekkel. Sajnos az elnevezések nem egyértelműek, mert gyakran ,,application virtualization" néven említik mindkétféle megoldást. Ennek oka részben abban keresendő, hogy számos futtatókörnyezet mellékesen nyújt támogatást alkalmazás izolációra illetve telepítés nélküli önhordó csomagból indítható alkalmazásokra is (pl. Java WebStart).

- Flexibilitás

Gyakori, hogy a hardverek teljesítményét, erőforrásait nem lehet optimálisan kihasználni, mert az alkalmazások *funkcionális* és *„nemfunkcionális”*[3] (más szóval *extrafunkcionális*) követelményei megkötéseket támasztanak azzal kapcsolatban, hogy miket telepíthetünk egy gépre. Például különböző operációs rendszeren futó szolgáltatásokat nem telepíthetünk egyazon gépre. Ez nemcsak szerverek, hanem asztali gépek, munkaállomások esetén is gyakori probléma, például többplatformos szoftverfejlesztésnél. Biztonsági szempontból kritikus alkalmazást sem célszerű telepíteni olyan gépre, ami az Internet irányába nyitott szolgáltatást nyújt. Számos nagyvállalati szoftver olyan követelményeket támaszt az operációs rendszer környezetével szemben, ami megakadályozza, hogy más szoftverekkel együtt fusson egy gépen. Az olyan operációs rendszerek menedzsmentje eleve nehézkessé válik, amin túlságosan sok és sokféle szolgáltatás fut egyszerre. Ezen követelmények miatt gyakran kell külön gépre telepíteni olyan alkalmazásokat, amik nem használják ki, vagy csak az idő kis részében használják ki a hardver kapacitását. Ezért érdemes lehet a "külön gép" és a "külön hardver" fogalmát elválasztani, és a hardverek szükségtelenül nagy számát konszolidálni.

A virtualizáció lehetővé teszi, hogy szeparált környezeteket alakítsunk ki, így egy gépre összevonhassunk olyan szolgáltatásokat, amiket a fent felsorolt vagy ehhez hasonló okok miatt dedikált gépekre kellene telepíteni. Ezzel a hardver kihasználtsága javítható, ezáltal a szolgáltatás kiépítési és üzemeltetési költsége (jellemző mérőszám a *Total Cost of Ownership*, *TCO*) jelentősen csökkenthető. A rendelkezésre állás javítható, hiszen egy meghibásodás a virtuális gépen belül elszigetelődik, a fizikai gépen futó egyéb szolgáltatásokat nem érinti. Hasonlóképpen a biztonsági kockázat is csökken, mert az esetleges illetéktelen hozzáférés is egy-egy virtuális gépen belülre korlátozódik (mMeg kell jegyezni, hogy a virtualizációs keretrendszerben is lehetnek biztonsági hiányosságok, az ilyen hibák viszont lényegesen ritkábbak az operációs rendszeren belüli hibáknál).

Nagy számú gép karbantartása és változó igények esetén új szerverek üzembeállítása, ideiglenes tesztkörnyezetek kialakítása gyakori feladatok a nagyvállalati, de sok esetben a kis- és középvállalkozások informatikai rendszereiben is. A virtualizáció lehetővé teszi a dinamikus erőforrás-allokációt, ami megkönnyíti új gépek üzembe állítását. További fontos szempont a távmenedzsment. Léteznek megoldások számítógépekhez való távoli hozzáférésre, ám ezek sokáig csak drága felső kategóriás szervergépekben voltak megtalálhatóak. Ezen megoldások hatóköre is korlátozott, vannak olyan feladatok, amik nem végezhetők el fizikai hozzáférés nélkül (konfiguráció megváltoztatása). A virtualizáció, lévén, hogy szoftveresen hozza létre a virtuális környezetet, lehetőséget biztosít arra, hogy a virtuális gépeken olyan műveleteket is elvégezzünk távolról, amiket fizikai gépen csak fizikai hozzáféréssel tehetünk meg.

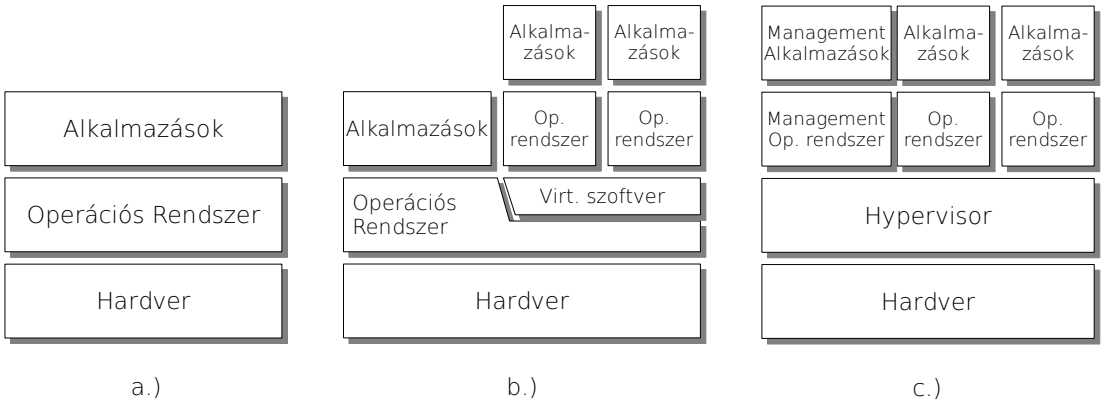
Fontos megjegyezni, hogy a virtualizáció minden esetben teljesítményveszteséggel (virtualizációs *overhead*) jár, ez bizonyos esetekben elenyésző, ám vannak olyan szélsőséges esetek is, ahol a virtualizált rendszer teljesítménye csak 10%-a a közvetlen hardveren való futtatás teljesítményének.

3. Platformvirtualizációs megközelítések

Teljes számítógép virtualizálására két architektúra terjedt el, a 2. ábra ezeket hasonlítja össze a megszokott virtualizációmentes esettel (2.a ábra).

Az ún. *hosted* virtualizáció (2 b. ábra) esetén a hardver erőforrásait egy általános célú operációs rendszer (a továbbiakban *host*, gazda operációs rendszer) kezeli, és efelett fut egy *virtualizációs szoftver*, ami futtatja a virtuális gépeket (a továbbiakban vendég, *guest* gépek), és biztosítja a virtuális hardver környezetet. Ilyenek például a VMware Workstation, Player termékei illetve a VirtualBox, Microsoft Virtual PC és a Linux KVM. Ennek előnye, hogy viszonylag könnyen telepíthető meglévő rendszerekre. A hardver eléréséhez a gazda operációs rendszer meghajtóprogramjait (*driver*) veszi igénybe, mint bármely más alkalmazás. Számos

[3] A teljesítmény is egy jellegzetesen nemfunkcionális követelmény, itt most a teljesítménynél fontosabbnak ítélt nemfunkcionális követelményekről van szó, pl. biztonságról.



2. ábra. Platformvirtualizációs architektúrák

operációs rendszer szintű erőforrás biztosítható a virtuális hardver számára, így egyszerűen megoldott a virtuális gépek kezelése, mert a gazda operációs rendszeren ablakban megjeleníthető a képernyőkép, hozzáférhet a gazdagép fájlrendszeréhez stb. Létezik olyan megoldás is (Paralells seamless windowing, VMware Unity), amely a vendéggépen futó alkalmazások ablakait a gazdagép grafikus felületén natív ablakokba helyezi, így a felhasználó elől elrejti, hogy valójában az alkalmazás virtualizált környezetben fut. Ezen kényelmi szolgáltatások miatt főleg asztali alkalmazások vagy munkaállomások esetén alkalmazzák hosted megoldást. Hátránya viszont, hogy a plusz réteg miatt a teljesítményveszteség nagyobb lehet. Sok virtuális gép futtatása egy gazdagépen általában problémás, kevés olyan optimalizálási lehetőség van, mely sok gép hatékony futtatását segítené.

A másik architektúra az ún. *bare-metal* virtualizáció (2 c. ábra), *hypervisor* használatával. A hypervisor lényegében egy speciális operációs rendszer kernel, amely virtuális hardverkörnyezetet biztosít. Minden virtuális gép e felett fut, sokszor még a hypervisor menedzseléséhez használt operációs rendszer is valójában egy virtuális gép. A hypervisor is rendelkezhet saját meghajtóprogramokkal, de létezik olyan megoldás is, amikor a vendég operációs rendszer közvetlen hozzáférést kap valamely perifériához. Általában a menedzsment operációs rendszer speciális abból a szempontból, hogy számos hardverhez nem virtualizált, hanem közvetlen hozzáférése van. Ennek a megközelítésnek előnye, hogy a hypervisor saját CPU és IO ütemezőt használhat, nem kell illeszkednie egy gazda operációs rendszerhez, annak memóriakezeléséhez és erőforrásgazdálkodásához. Ezáltal lehetővé válik például a virtuális gépek memóriatartalmában a közös részek osztott memóriába összevonása, a processzorok rögzített hozzárendelése egy-egy virtuális géphez illetve a részletes erőforráshasználat-korlátozás.

Hypervisor architektúrát használ pl. a Xen, a Microsoft Hyper-V valamint a VMware ESXi[4]. Ez egy sajátos megközelítés, ahol a menedzsment alkalmazások külön-külön mintegy saját virtuális gépben (ún. *world*-ben) futnak a hypervisor kernel felett.

3.1. Emuláció, hardveres- és paravirtualizációs megoldások

Ahhoz, hogy a vendég operációs rendszert ne befolyásolja, hogy valódi hardver helyett virtualizált környezetben fut, a futtatórendszernek a speciális (*privilégizált*, csak operációs rendszer kernel által használható) utasításokat el kell fognia (*trap*), és ki kell cserélnie saját rendszerhívásaira. Az x86-os architektúra jónéhány olyan privilégizált utasítással rendelkezik, amire nem lehet hardveres elfogást beállítani. Hasonló a probléma a memóriakezeléssel, hiszen az operációs rendszerek a virtuális memória lapok (*page*) hatékony kezelésekor hardveres megoldásra támaszkodnak (MMU), a virtuális gépek számára azonban egy indirekciós réteget kell beiktatni. Ennek a problémának a megoldására három fő megoldási irány létezik:

[4]A VMware ESXi a hypervisor egy olyan változata, ahol megszüntették a dedikált menedzsment operációs rendszert.

- **Emuláció** – egy futtatószoftver vizsgálja, és alakítja át a vendég operációs rendszer által végrehajtott utasításokat. Saját maga tart karban egy legfelső szintű laptáblát a memóriaelérések kezelésére. Ez a leglassabb, de a leginkább flexibilis megoldás, mert ez lehetővé teszi, hogy a fizikai processzorétól eltérő utasításkészletet használjon a virtuális gép. Ilyen megoldás a QEMU. Platformvirtualizációnál a viszonylag rossz teljesítménye miatt kívánatos elkerülni az emuláció alkalmazását. *Szoftveres virtualizációnak* nevezzük az emulációnak azon speciális formáját, amikor azonos a fizikai és a virtualizált platform, és csak bizonyos utasításokat kell újrafordítani, más utasítások változtatás nélkül végrehajthatóak. Ez a megoldás a tiszta emulációnál lényegesen jobb teljesítményt nyújthat.

- **Paravirtualizáció** – a vendég operációs rendszert módosítjuk, hogy ne hajtson végre olyan utasításokat, amik problémát okoznak, helyettük használjon egy-egy speciális rendszerhívást. Hasonlóképpen a memória laptáblák kezelésére is speciális rendszerhívásokat használ a közvetlen MMU elérés helyett. A vendég operációs rendszer CPU ütemezése is kooperálhat a virtualizációs keretrendszer ütemezőjével. Ám ez sok esetben nem használható, mert a vendég operációs rendszerben nem lehet módosításokat végezni. A nyílt forrású (mindenekelőtt a Linux alapú) rendszerekben ez megoldott, a legújabb kernel kiadások forráskódjában már eleve benne vannak a paravirtualizációt támogató részek. Windows esetén általában nem lehet tisztán paravirtualizációt megvalósítani, mert a kernel nem módosítható, ilyenkor a CPU emulációnak és a perifériák paravirtualizációjának kombinációját alkalmazzák, ami szintén viszonylag jó teljesítményt biztosíthat.

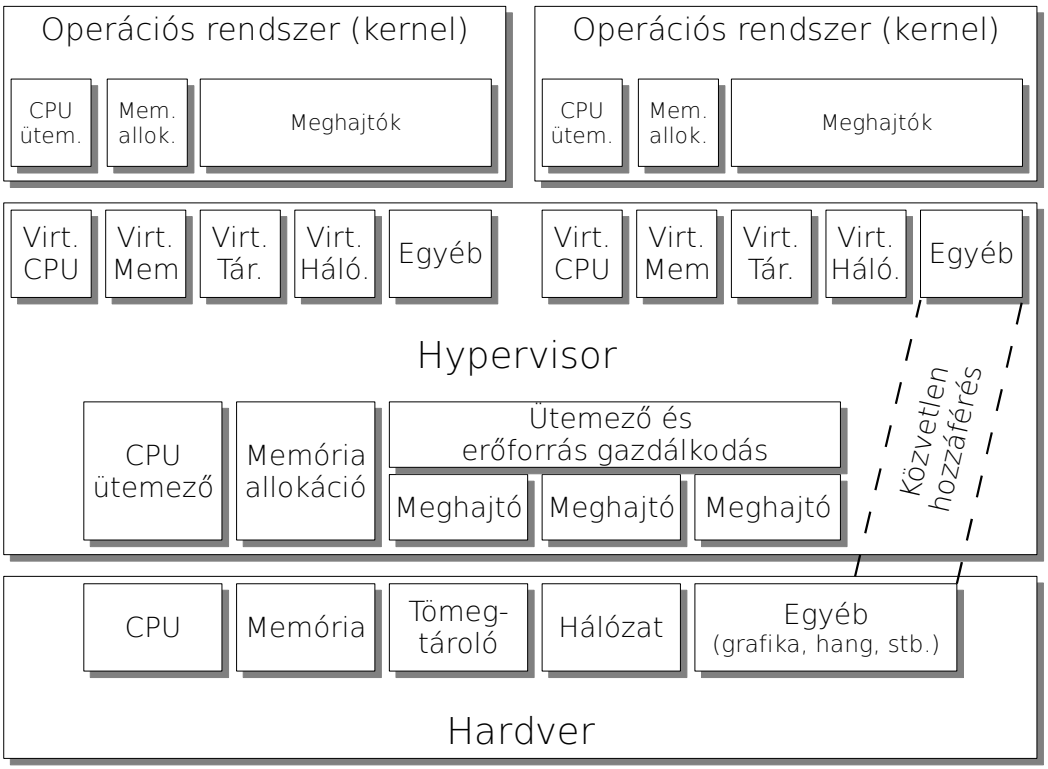
- **Hardveres virtualizáció** – a CPU utasításkészletének olyan kiegészítése, ami lehetővé teszi a vendég operációs rendszer kódjának módosítás nélküli futtatását. Az Intel és az AMD újabb processzorai tartalmaznak ilyen kiegészítéseket (Intel VT, AMD-V), ám régebbi processzorokkal szerelt gépek esetén ez a megoldás nem alkalmazható. Nem minden virtualizációs környezet képes kihasználni ezt a lehetőséget, ez főként a régebbi verziójú, 2006 előtti VMware, VirtualBox, MS VirtualPC illetve Parallels kiadásokra igaz. Léteznek olyan megoldások is, amelyek viszont teljes egészében erre épülnek, így a CPU utasításkészlet kiegészítése nélkül nem is működőképesek. Ilyenek például a Linux KVM, a Microsoft Hyper-V valamint a Windows 7 XP kompatibilitási virtuális gépe.

A VMware termékei szoftveres virtualizációra alapulnak, ami kiegészülhet paravirtualizációval (VMI paravirtualization), de egyes esetekben (pl.: 64 bites vendég futtatása) hardveres virtualizációt igényelnek. Más szoftverek (pl. Xen) tisztán paravirtualizációt használnak, ha a vendég operációs rendszer ezt lehetővé teszi, egyébként hardveres (natív) virtualizációra hagyatkoznak. Ha a processzorból hiányzik az ehhez szükséges kiegészítés, teljes emulációra lépnek vissza.

3.2. Perifériák virtualizálása

A vendég operációs rendszerek számára biztosított hardver a processzoron és az allokált memórián kívül tartalmaz virtuális merevlemezt és hálózati interfészt, grafikus megjelenítőt, opcionálisan egyéb perifériát is. Ez leggyakrabban soros, párhuzamos vagy USB port, néhány rendszer esetén hangkártya vagy akár grafikus gyorsító is lehet. A virtuális hardver többféleképpen lehet megvalósítva. A főbb lehetőségek hasonlóak az alapvető virtualizációs megoldásoknál tárgyaltakhoz:

- **Emulált periféria** – valamilyen létező hardver programozói felületét (regiszterkészletét, interrupt és DMA viselkedését) egy szoftveres implementáció valósítja meg a valódi hardvert a lehető legpontosabban utánozva. A vendég kernel driverei pontosan úgy működnek, mintha valódi hardvert kezelnének, miközben valójában a hardvert emuláló szoftverrel kommunikálnak. A gyakorlatban ez a legkevésbé problémás megoldás, mert ha a virtualizációs keretrendszer egy kellően elterjedt hardvert emulál, akkor a vendég operációs rendszer azt kezelni fogja a beépített meghajtóprogramjaival. Ez főleg a vendég operációs rendszer telepítése során fontos, amikor még nincs lehetőség saját meghajtókat betölteni. Ezt a megoldást implementálni elég körülményes, meglehetősen nagy CPU terheléssel jár, viszont az



3. ábra. Periféria virtualizációs lehetőségek összefoglalása

emulált hardver eltérő lehet a fizikaitól. Akár operációs rendszer szintű erőforrás is könnyen használható virtuális hardverként, például egy fájl tartalma merevlemezként.

- **Paravirtualizált perifériák** – leegyszerűsített programozói interfész használata a hardver emulátorban, az egyes I/O műveletek helyettesítése rendszerhívásokkal. Így gyakorlatilag a regiszterek, direkt memória-hozzáférés és megszakítások emulációja elhagyható, a rendszerhívás-interfészen keresztül a magasszintű műveletek lényegében függvényhívásszerűen zajlanak le. Ez a megoldás nagyon gyors, hátránya azonban, hogy saját meghajtóprogramok telepítését igényli, hiszen a valóságban nem létezik olyan hardver, amelyiknek a programozói felülete hasonlítana a paravirtualizált hardverekéhez. Ennél a megoldásnál van a legtöbbféle lehetőség operációs rendszer szintű erőforrások virtuális hardverként való kiajánlására (pl. fájlrendszer is kiajánlható, nemcsak blokkos eszköz).

- **Közvetlen hardverelérés** – az egyes hardverelemek dedikáltan hozzárendelhetőek egy-egy virtuális géphez, amelynek az operációs rendszere közvetlenül vezérli a hardvert. Ez egy rendkívül gyors és hatékony módja a virtuális perifériák megvalósításának, ám sok szempontból kompromisszumot jelent. Jelenleg ez leginkább azt jelenti, hogy egy hardvert legfeljebb egyetlen virtuális gép vezérelhet egyidejűleg, a hozzárendelés statikus. Csak a tényleges fizikai hardverfajta virtualizálható, a vendégnek ezt kell támogatnia. A szeparációt is aláaknázhatja ez a megoldás, mert sok olyan hardverfajta van, ami felprogramozható, hogy tetszőleges *fizikai* címen végezzen DMA műveleteket. DMA segítségével (akár nem szándékosan vagy meghibásodásból kifolyólag is) hozzáférhet egy virtuális gép olyan memóriaterülethez, ami más virtuális géphez, a gazdagéphez vagy esetleg a hypervisorhoz tartozik. Ezt a megoldást hátrányai miatt általában kerülni szokták, csak speciális esetekben alkalmazzák, különösen akkor, ha olyan hardvert kell elérhetővé tenni, amihez a virtualizációs keretrendszernek nincs drivere (pl. 3D grafikus gyorsítók). Új fejlesztés az

IOMMU-val szerelt alaplapi chipsetek (Intel VT-d) és többszörös konkurens hozzáférésre felkészített perifériák megjelenése, mely kiküszöböli ennek a megoldásnak számos hátrányát.

4. A VMware ESXi bemutatása

A VMware cég a nagyvállalati szervervirtualizációs megoldását forgalmazza *vSphere* platfrom néven. Ennek alapkomponense az **ESXi Server**, mely egy **hypervisor** alapú virtuális gép futtatókörnyezet. A korábbi, ESX nevű változat tartalmazott egy menedzsment operációs rendszert, ami egy módosított RedHat Enterprise Linux 3. Az ESXi-ben ez a Linux-alapú menedzsment operációs rendszer nem található meg, így az alap rendszer lényegesen egyszerűbbé vált, és az erőforrás-fogyasztása is csökkent. Az alap rendszer akár USB flash eszközre is telepíthető, valamint hálózatról is bootolható. Létezik olyan szerver hardver, melybe rendelhető beépíthető pendrive-on ESXi hypervisor, így az teljes egészében merevelemez nélkül használható. Ilyenkor a virtuális gépek valamilyen tárhálózaton foglalnak helyet.

A mérési környezetben már az ESXi változat található meg, melynek a teljes bootolása hálózatról történik.

4.1. Az ESXi szolgáltatásai

Az ESXi célja, hogy kevés kiépítési és karbantartási ráfordítással biztosítson nagyvállalati igényeket is kielégíteni képes szervervirtualizációs környezetet az ehhez szükséges összes kiegészítő szolgáltatással (pl. távoli elérés, erőforrások dinamikus kiosztása és átkonfigurálása, hozzáférési jogosultságok kezelése stb.) együtt. Fontos megjegyezni, hogy a VMware a teljes termékskálájában kompatibilis virtuális gép formátumot használ. Ez azt jelenti, hogy (néhány megkötéssel) a Workstationnel készített virtuális gépek futtathatóak pl. Playerrel ESXi-n is, és fordítva.

4.2. Az ESXi felépítése

Az ESXi szerveren a menedzsment funkciókat, távoli elérést szolgáltató folyamatok a VMkernel nevű hypervisor rendszermag felett, a virtuális gépek mellett futnak. A korábbi ESX-hez hasonlóan itt létezik egy parancssori felület, de ennek használata csak bizonyos esetekben támogatott.

Közvetlenül a gép konzolján csak néhány alap beállítás végezhető el (jelszó beállítása, hálózat kiválasztása, gép újraindítása). Minden egyéb művelet, beleértve a gazdagép további konfigurálását valamint a virtuális gépek létrehozását, futtatását, távoli eléréssel végezhető el.

Van egy webes kezelőfelület, amivel meg lehet tekinteni a konfigurációt, és néhány alapműveletet el lehet végezni a virtuális gépeken. Az ESXi webszolgáltatásokon (Web Services) keresztül biztosít kívülről elérhető programozói felületet, továbbá támogatja a WBEM (Web Based EnterpriseManagement) és CIM (Common Information Model), WS-MAN (Web Services for Management) illetve SNMP (Simple Network Management Protocol) szabványokat is.

A vSphere Client egy Windows-on (tehát külön gépen) futó vastagkliens alkalmazás, mellyel az ESXi szerverekhez kapcsolódhatunk a webszolgáltatásos felületen. Ennek segítségével konfigurálhatjuk a gazdarendszert, illetve kezelhetjük a virtuális gépeket.

Ez kiegészülhet még a **vCenter Server**rel, ami egy Windows Serveren futó komponens, amely számos ESXi host kezelését képes központosítottan megoldani, és számos fejlettebb menedzsmentképességgel terjeszti ki azokat. Ehhez is vSphere Client alkalmazással kapcsolódhatunk.

4.3. Főbb alrendszerek

Virtuális tár – a virtuális gépeket leíró konfigurációs fájlok (`.vmx`, `.vmxf`), a virtuális gép állapotát tároló fájlok (`.nvram`, `.vswp`) és a virtuális gép merevlemezeinek tartalmát tároló fájlok (`.vmdk`) tárolására szolgál. Az ESXi egy speciális fájlrendszert (*VMFS*) használ erre a célra, melyet kifejezetten nagy fájlok tárolására optimalizáltak. Valójában alacsony szinten a logikai kötetkezelőkhöz hasonló elven működik, ám egy szabályos fájlrendszer felületet biztosít, ami a kezelését egyszerűsíti. Fontos tulajdonsága, hogy fürtözött (clustered) elérést is lehetővé tesz, tehát egy hálózati tárrendszeren (SAN) elhelyezve egyszerre több különálló ESXi is használhat egyazon közös tárat. Hálózati tárolást a VMkernel biztosít, ami egy teljesen független hálózati stack és hálózati tár elérési protokoll implementációval rendelkezik.

Virtuális hálózat – a virtuális gépek hálózati kapcsolatát virtuális switch-ek (gyakorlatilag szoftveres Ethernet hidak) biztosítják. A virtuális switch-ek egy közbenső réteget jelentenek a fizikai interfészek és a virtuális gépek hálózati interfészei között, használatukkal flexibilis hozzárendelést lehet kialakítani. A VMkernel hálózati hozzáférését is virtuális switch-eken konfigurálhatjuk, ezeknek is külön MAC címük és IP címük van, pontosan úgy, mintha fizikai hálózati interfésszel rendelkeznének. Ezáltal megoldható, hogy egyetlen fizikai hálózati interfészen menjen a hálózat tárrendszer és a virtuális gépek forgalma. Lehetőség van zárt (gazdagépen belüli, fizikai interfész nélküli) virtuális hálózatok definiálására is, ami csak egyetlen gazdagépen futó több vendéggép között működik.

CPU kezelő és ütemező – a gazda CPU idejét osztja szét a vendéggépek és a VMkernel felett futó szolgáltatások között. Lehetőség van prioritások és minimálisan garantált illetve maximális kioszthatő CPU idő definiálására is. Az ütemező felismeri a processzorok topológiáját, a többmagos processzorokat, a Hyperthreading technológiát. Figyelembe veszi az ütemezéskor, hogy mely logikai processzorok függetlenek egymástól, mennyi költséggel jár egy virtuális gépet, ami idáig az egyik processzoron futott, átütemezni egy másik processzorra. Ez különösen nagyméretű, nem egységes memóriaelérést használó rendszereknél (NUMA, Non Uniform Memory Access) számít teljesítményben sokat. A CPU virtualizáló rendszer működhet szoftveresen virtualizált, paravirtualizált és hardveres virtualizált üzemmódban. A szoftveres virtualizált mód 32 bites vendég operációs rendszereknél aktív. Ha lehetséges, paravirtualizációs üzemmódba vált át, ezt jelenleg külön engedélyezni kell a virtuális gép CPU tulajdonságainál. 64 bites vendég operációs rendszereknél hardveres virtualizációt használ, ennek hiányában nem lehet 64 bites virtuális gépet futtatni.

Memóriakezelő – a CPU ütemezőhöz hasonlóan ez is kezel minimális és maximális foglalási beállításokat valamint prioritásokat. Dinamikus memóriakiosztást használ, ami virtualizáció esetén a hagyományos operációs rendszerekétől némileg eltérő jellegű feladat, mert a virtuális gépeknek adott memóriaméretük van, amiről feltételezik, hogy az számunkra rendelkezésre áll. A VMkernel memóriakezelője figyeli az egyes virtuális gépek aktivitását és memórialap-használatát, ennek megfelelően képes késleltetni a memória tényleges allokálását egy virtuális gép számára. Ilyen módon lehetőség nyílik akár a gazdagép túlfoglalására (overbooking) is, bár ez erősen ellenjavallt. Ha a gazdagép memóriája nem elég, akkor képes lapozófájlba (swap) írni a virtuális gépek memóriatartalmát. A teljesítményre kifejezetten káros lehet, ha a virtuális gép operációs rendszere lapoz (belső swap), miközben a virtualizációs keretrendszer is lapozza memóriáját (külső swap), mert a vendég operációs rendszer nem tudja, hogy az ő általa gyors memóriának gondolt címtartomány egy része is háttértáron van, így előfordulhat, hogy a két swap terület között ideoda másolgatják a memórialapokat, ami vergődést (trashing), azaz a háttértár alrendszer nagy terhelése melletti rendkívül alacsony teljesítményt okoz. Ennek elkerülésére létezik paravirtualizációra alapuló kooperációs technika a kernel és a vendég operációs rendszer között (*memory balooning*).

Ezen túl még van számos, nem kevésbé jelentős alrendszer is, pl a vendéggépek grafikus felületéhez való távoli hozzáférést biztosító virtuális framebuffer, amit itt nem tárgyalunk részletesen.

4.4. A virtuális hardver felépítése

A virtuális gépek számára biztosított hardverkörnyezet tartalmaz[5]:

- **CPU** – 1, 2 vagy 4 db lehet (de nem több, mint a gazdagépben lévő processzorok száma), típusa azonos a gazda CPU-éval, ám opcionálisan letilthatóak speciális kiegészítések (SSE, 64bit, Nx bit stb.). Ez utóbbira akkor van szükség, ha működő (vagy felfüggesztett) virtuális gépet akarunk mozgatni eltérő CPU-val szerelt gazdagépek között. Az ilyen letiltás teljesítményvesztéssel járhat, és még a virtuális gép elindítás előtt kell beállítani.

[5] Az itt megadott korlátok ESXi 4 esetén érvényesek.

- **Memória** – Maximum 64 GB memória osztható ki, akár 32 bites vendégeknek is, emulált PAE (Paging Address Extension) kiterjesztéssel. A memóriaméret is csak leállított virtuális gépen módosítható.
- **Chipset** – Intel 440BX chipsetet emulál. Csak megszakítás és periféria erőforrás konfigurációs szerepe van.
- **BIOS, NVRAM** – Egy saját BIOS-t biztosít. A BIOS beállításokat tároló nem felejtő memória tartalmát külön fájlban (`.nvram`) tárolja.
- **Grafikus vezérlő** – Alap VGA, VESA üzemmódokat emulál, ez az üzemmód nagyon lassú. Létezik paravirtualizált üzemmódja is, mely már jól használható sebességet biztosít.
- **IDE vezérlő** – Az ESXi csak CD/DVD meghajtót emulál IDE felett, a Workstation merevlemezt is támogat.
- **SCSI vagy SAS vezérlő** – LSI Logic vagy Buslogic típusú emulált hardver, külön meghajtóprogram van hozzá, amit telepítésnél pl. a Windows XP-nek meg kell adni.
- **Floppy vezérlő** – Egy floppy lemez képét tartalmazó fájlból szolgáltat tartalmat, vagy a gazdagép fizikai meghajtóját kapcsolja hozzá a virtuális géphez. Lehetőség van a vSphere klienst futtató gép meghajtójának csatlakoztatására is hálózaton keresztül.
- **CD/DVD** – Egy CD vagy DVD képét (`.iso`) tartalmazó fájlból szolgáltat tartalmat, vagy a gazdagép fizikai meghajtóját kapcsolja hozzá a virtuális géphez. Itt is lehetőség van a vSphere klienst futtató gép meghajtójának csatlakoztatására hálózaton keresztül.
- **Merevlemez** – ESXi alatt csak a SCSI vezérlőre csatolható. Tetszőleges méretű fájl (`.vmdk`) hozható létre az adatok tárolására. Fontos, hogy az ESXi alapértelmezetten a teljes lemezt allokálja (*flat*, *thick* formátum), a dinamikusan növekvő helyfoglalást biztosító (*thin*) formátumot külön kell kiválasztani. Nem támogatott a Workstation és a Player *2gbsparse* formátuma, mely 2GB-os fájlokra darabolja a virtuális lemezt. Az ilyen lemezképeket konvertálni kell.
- **Hálózati interfész** – Lehetőség van AMD PCNet32 vagy Intel E1000 (Intel PRO/1000 termékcsalád) kártyák emulálására vagy VMXNet meghajtóval paravirtualizált Ethernet interfész kialakítására. Az éppen használt típust többnyire automatikusan választja ki („`flexible`”), azonban ez manuálisan felülbírálható.
- **Soros port** – A gazdagép soros portjára, egy fájl tartalmára vagy hálózaton keresztül a klienst futtató gép portjára kapcsolható.
- **Párhuzamos port** – A gazdagép párhuzamos portjára, egy fájl tartalmára vagy hálózaton keresztül a klienst futtató gép portjára kapcsolható.
- **Egyebek** – Csak Workstation alatt támogatott: **Audio vezérlő**, **USB port**, **Megosztott könyvtár** (fájlrendszer megosztás kiajánlása).

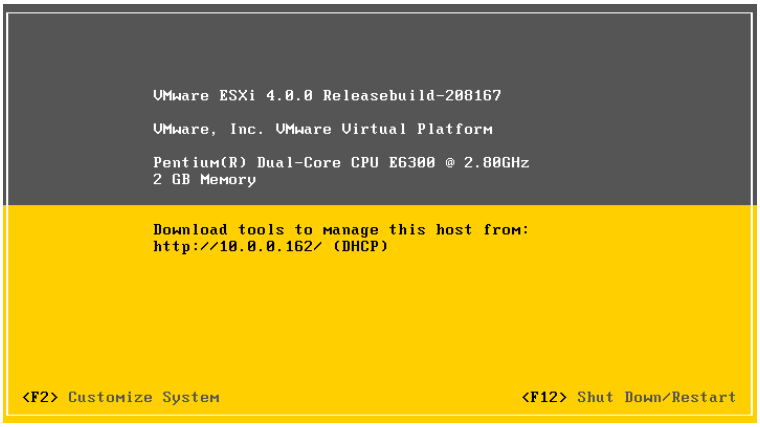
Mint már említettük, a jó teljesítmény elérése érdekében a paravirtualizáció használata kívánatos. Perifériák paravirtualizálására szolgál a **VMware Tools** csomag, ami paravirtualizált meghajtóprogramokat tartalmaz a vendég operációs rendszer részére. Leginkább a grafikus vezérlő sebességén és az egér transzparens kezelésén (simább mozgás, ablak szélén be- és kilépés) vehető észre a Tools hatása, de általában minden virtuális hardverkomponens teljesítményén jelentősen javít.

5. ESXi kezelése a vSphere Clienttel

Az ESXi 4.0 verziójához a vSphere Client (régebbi nevén Virtual Infrastructure Client) vastagkliens alkalmazással lehet kapcsolódni. A vSphere Client telepítőkészlete letölthető az ESXi webes felületéről.

A laborban a gépeken nincs ESXi telepítve, hanem az a hálózatról bootolható üzemmódban használható. Ez egy hivatalosan nem támogatott megoldás, de néhány részlettől eltekintve teljesértékű működést biztosít. A legfontosabb hiányosság ebben az üzemmódban az, hogy újraindítás után elveszíti a beállításait. A mérési feladatok megoldása során várhatóan nem lesz szükség újraindításra.

Egy frissen telepített vagy hálózatról indított gazdagépre távolról még nem lehet belépni, először a `root` felhasználónak jelszót kell beállítani. Ezen kívül érdemes ellenőrizni, hogy a gép hálózati kapcsolata működik-e. Az ESXi alapértelmezetten DHCP-vel automatikusan IP címet kér. A kapott IP cím a szöveges konzolon leolvasható (4. ábra). A jelszó valamint a hálózati beállítások menüpontjai az <F2> lenyomásával érhetőek el (5. ábra).

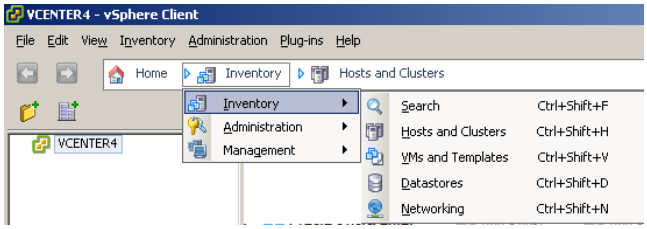


4. ábra. Az ESXi 4.0 konzol képernyője

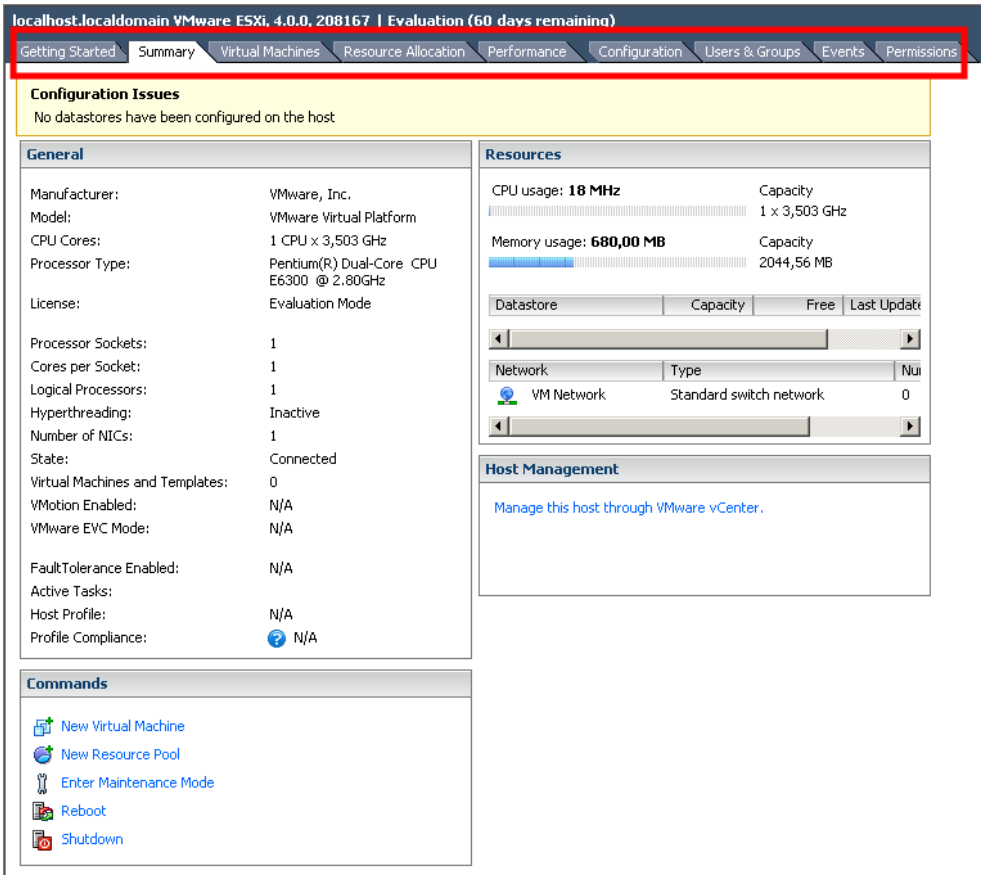
5. ábra. ESXi alapbeállítások

Távoli gépről a *vSphere Client* alkalmazással lehet belépni a konzolon megadott jelszóval. Belépés után a kezelőfelületen felül a menüsor alatt található a nézetkiválasztó sáv (6. ábra). A nézetek jelentős része csak akkor érhető el, ha vCenter Serverhez kapcsolódunk. Bal oldalt az *Inventory* nézetben, egy erőforrásokat nyilvántartó fa található, legfelül a gazdagéppel. Jobb oldalon az aktuálisan kiválasztott objektum (gazdagép, erőforrás-csoport, virtuális gép stb.) tu-

lajdonságait tekinthetjük meg, illetve állíthatjuk be különböző füleken (7. ábra). Fontos kiemelni, hogy a továbbiakban minden beállítás és művelet a menedzselt ESXi futtató gépre vonatkozik, nem pedig arra, amelyiken a kliensprogramot elindítottuk.



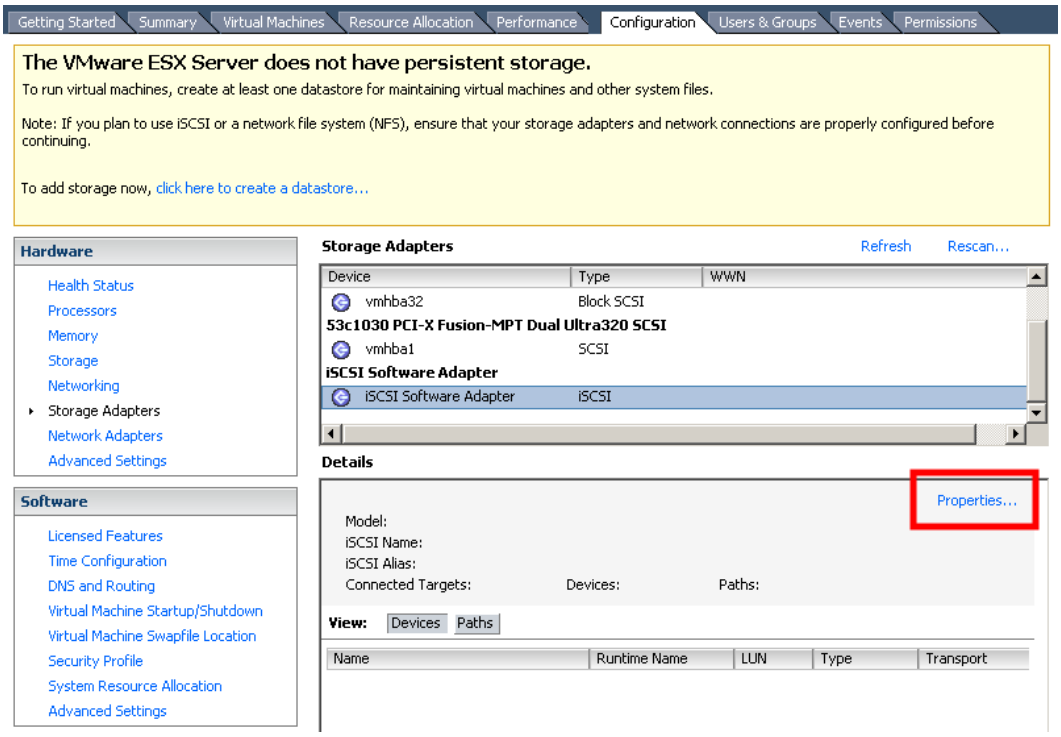
6. ábra. A vSphere Client nézetkiválasztó sávja



7. ábra. A vSphere Client menedzsmentfelületének különböző fülei

5.1. Az alapkonfiguráció beállítása

A gazdagépet kiválasztva a *Configuration* fülön végezhetjük el a beállításokat (8. ábra). A konfigurálható illetve megtekinthető beállításcsoportok bal oldalt találhatóak. Egy további általános megjegyzés a kezelőfelületen való eligazodás érdekében: az egyes beállításoknál a jobb oldali ábrák illetve listák fejlécében szereplő kék feliratok segítségével hívhatóak elő a beállítás módosító dialógus ablakok.

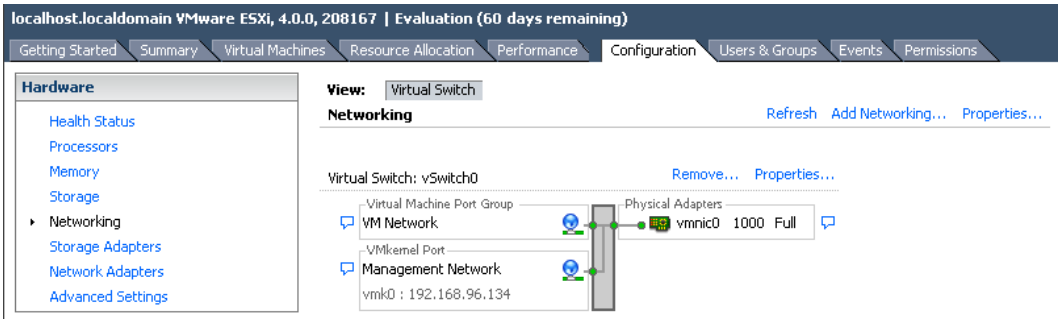


8. ábra. A menedzsmentfelület konfigurációs füle, háttértár adapterek

A frissen telepített vagy hálózatról indított ESXi példány még nem rendelkezik tárhellyel, a továbbiakban egy VMFS adattár iSCSI tárhálózaton történő létrehozásának lépései kerülnek ismertetésre.

Mivel az ESXi mostani változata a korábbi verziókkal ellentétben már támogatja a SATA merevlemezeket, ezért a laborgépek alap merevlemeze is fel fog tűnni a *Storage Adapters* valamint az *Add storage* menüpontban az eszközlistában. Ezeket NE használjuk, ellenkező esetben a tartalmuk felülíródik! Az iSCSI merevlemezek azonosítója `IET iSCSI Disk` lesz.

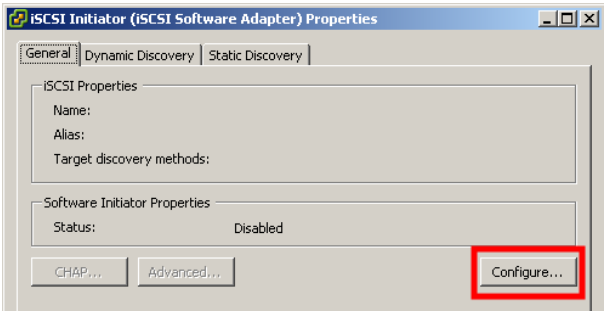
Az iSCSI konfigurálásához először meg kell győződni arról, hogy van-e VMkernel hálózati interfész hozzárendelve ahhoz a fizikai hálózathoz, melyen keresztül az iSCSI targetet el kívánjuk érni (9. ábra). Ennek különösen akkor van jelentőssége, ha egy szervergépben több hálózati interfész található, és a tárhálózathoz külön dedikált interfészt használunk (ahogyan ez általában szokásos).



9. ábra. A hálózatok összerendelésének ellenőrzése

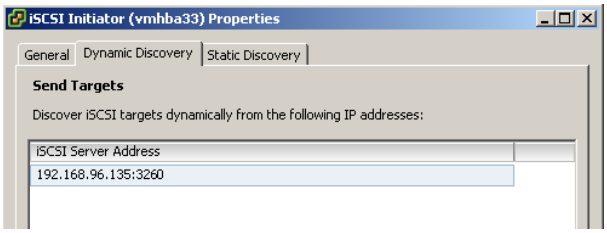
A *Storage Adapters* alatt (8. ábra) az *iSCSI Software Adapter* tulajdonságainál állíthatjuk be a célpontot. Az iSCSI-ról részletesebb ismertetés a „Háttértár rendszerek” mérés segédletében olvas-

ható. A *General* fülön a *Configure...* alatt engedélyezni kell az iSCSI kezdeményezőt (10. ábra).



10. ábra. Az iSCSI kezdeményező tulajdonságai

Az engedélyezés után még egyszer visszatérve a *Configure...*-ra átállíthatjuk az ESXi iSCSI állomásnevét, ám az alapértelmezett állomásnév megváltoztatása újraindítást igényel, erre most nem lesz szükség.



11. ábra. Az iSCSI célpont megadása

Az iSCSI célpont IP címének megadása a *Dynamic Discovery* fülön lehetséges (11. ábra). Új célpont hozzáadása után egy hosszabb várakozás következik. Lépjünk ki a dialógus ablakokból, és az *iSCSI Software Adapter*en indítsunk egy *Rescan...* műveletet. Ezt automatikusan is fel fogja ajánlani. Hagyjuk, hogy mindegyik felderítést elvégezze. A felderítés folyamatát az ablak alján lévő eseménynaplóban figyelhetjük. A felderítés végeztével az *iSCSI software adapter*t újra kiválasztva látnunk kell az iSCSI tárat a listában.

Ha a célponton már létre lett hozva VMFS fájlrendszer, akkor az meg fog jelenni az adattárak listájában a *Storage* fül alatt (12. ábra). Ha még nem, akkor létre kell hozni (*Add Storage...*). A varázslóban *Disk/LUN* típusú adattárat kell választani, majd ki kell választani az iSCSI merevlemezt a listából (13. ábra). Újabb általános megjegyzés a kezelőfelülettel kapcsolatban: minden varázsló végén van egy összefoglaló képernyő, ami az összes beállítást egy képernyőn mutatja, ezzel könnyítve az áttekintést valamint a műveletek dokumentálását (14. ábra).

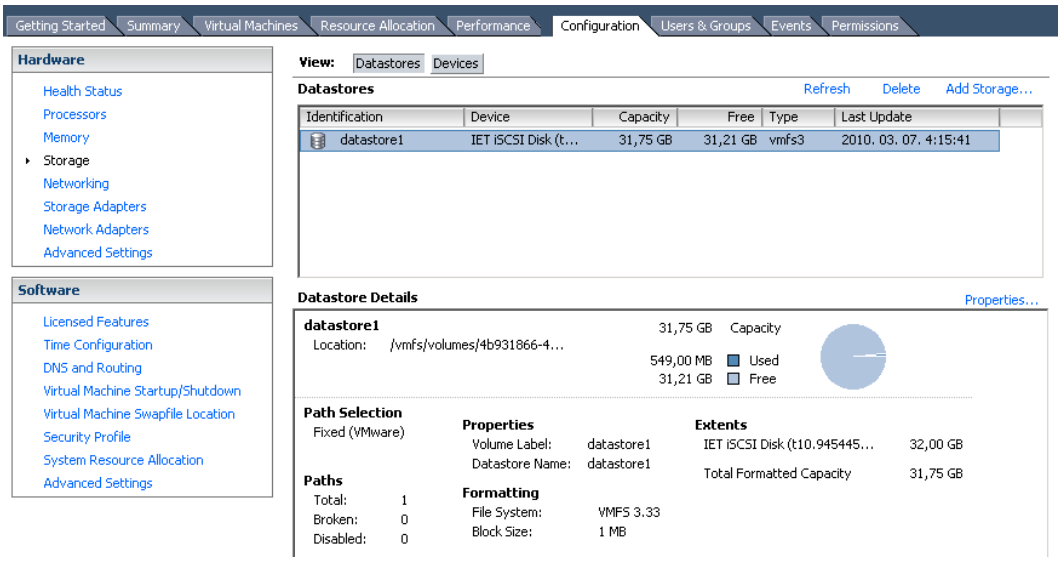
Innentől kezdve az ESXi rendelkezik adattárral, lehet virtuális gépeket felvenni.

5.2. Virtuális gépek létrehozása

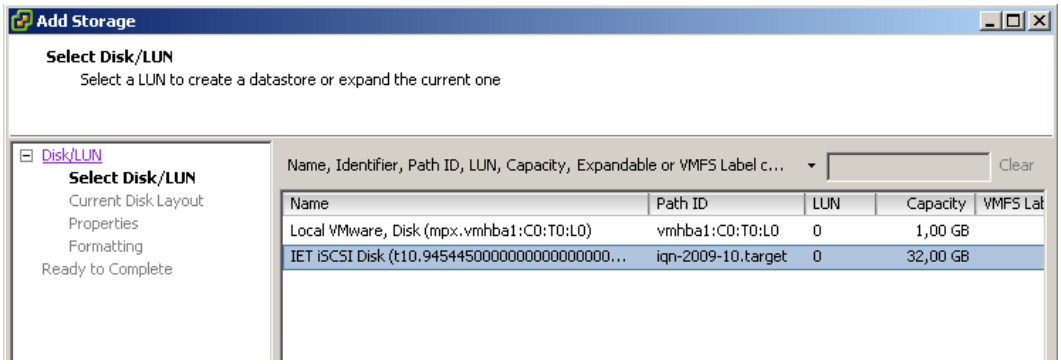
Az ESXi szerver felkonfigurálása után lehet új virtuális gépeket létrehozni vagy előregyártott virtuális gépeket letölteni, illetve ezután lehet új virtuális gépeket feltölteni az adattárba.

5.2.1. Új virtuális gép létrehozása

Az ESXi mindennapi használata során gyakori művelet, hogy új virtuális gépet hozunk létre. Ennek lépéssorozata sokban hasonlít a Workstation alatti lépésekhez. Lényeges eltérés, hogy a virtuális gépeket az erőforrás fában (*inventory*) helyezzük el.



12. ábra. Az adattárak listája

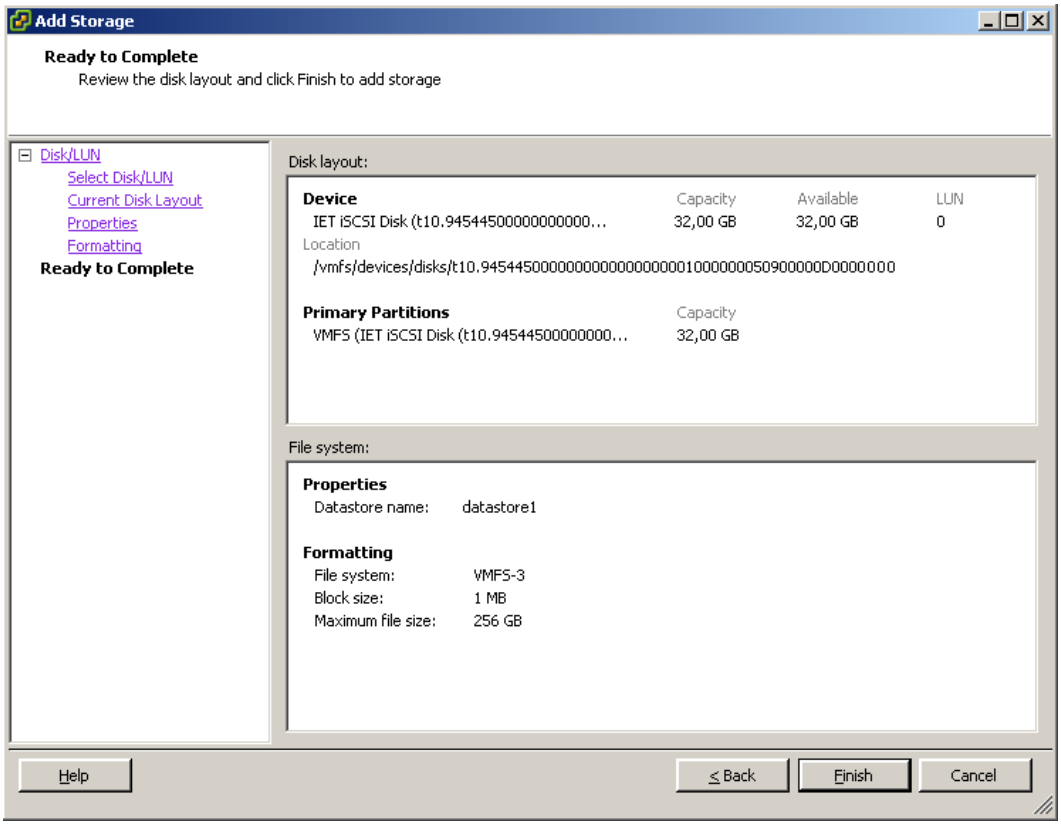


13. ábra. Az iSCSI merevlemez-lista

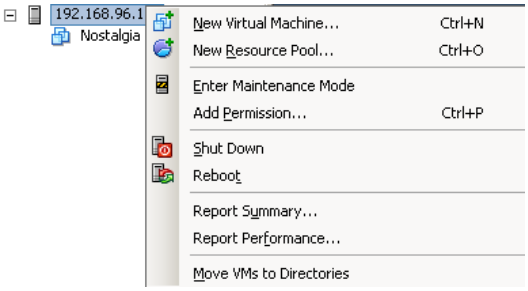
A virtuális gép létrehozását segítő varázslót (*New Virtual Machine Wizard*), a gazdagép összefoglaló (*Summary*) fülén vagy a gazdagépen jobb gombbal előugró kontextus menüből (15. ábra.) indíthatjuk.

A varázsló a következő beállításokon halad végig (16. ábra):

1. Virtuális gép neve
2. Adattár helye
3. Operációs rendszer típusa. Ez a támogatott virtuális hardverelemek és teljesítmény optimalizálása miatt fontos.
4. Processzorok száma
5. Memóriaméret
6. Hálózati interfészek és kapcsolódásaik a virtuális hálózati kapcsolókra
7. Virtuális lemez tárhely allokáció (itt választható ki a dinamikusan növekvő *thin* lemezformátum)



14. ábra. A háttértár konfigurálás összefoglaló képernyője



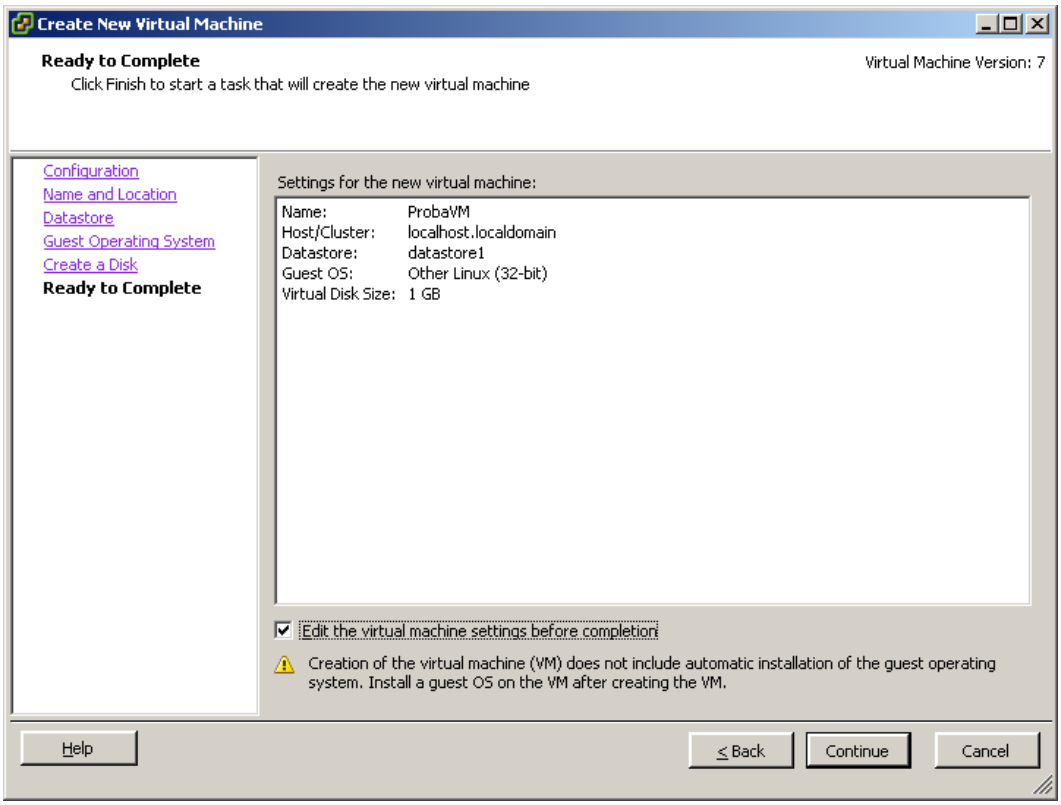
15. ábra. Egy gazdagép kontextus menüje

Ha a varázsló végén bejelöljük az *Edit virtual machine settings before completion* opciót, még a létrehozás előtt részletesen is módosíthatjuk a beállításokat. Már létező virtuális gépek esetén ugyanezt az ablakot az erőforrás fából a virtuális gép kontextus menüjéből vagy a *Summary* fülén az *Edit virtual machine settings...* menüponttal érhetjük el.

A beállításokat három fülre csoportosították: *hardware*, *options*, *resources*. Az első értelemszerűen a virtuális hardverelemek módosítására szolgál (17. ábra).

A második fül egyéb beállításokat tartalmaz, például a CPU- és a memóriavirtualizáció fajtáját, paravirtualizációt, a vendég gép BIOS-ába belépést (18. ábra, indítás után túl rövid idő állna rendelkezésre a megfelelő billentyű lenyomására), illetve a vendég operációs rendszer fajtáját.

Az utolsó fülön lehet a virtuális gép számára az erőforrás korlátokat, a garantált minimális erőforrás részesedést, valamint a részesedés prioritását megadni. (19. ábra)



16. ábra. Egy új virtuális gép beállítanivalói

5.2.2. Előregyártott virtuális gép letöltése

Manapság népszerű megoldás egy-egy feladat ellátására szolgáló virtuális gépet vagy egy-egy szoftver könnyen kipróbálható előretelepített és konfigurált változatát ún. virtuális „készülék” (*virtual appliance*) formájában terjeszteni. Ezzel kézi telepítés és kompatibilitási problémák nélkül, minimális beállításigénnyel juthatunk egy működő szoftverhez vagy szolgáltatáshoz. Ilyen készülékek terjesztésére egy szabványos formátum az OVF (Open Virtualization Format), melyet több gyártó terméke is képes fogadni.

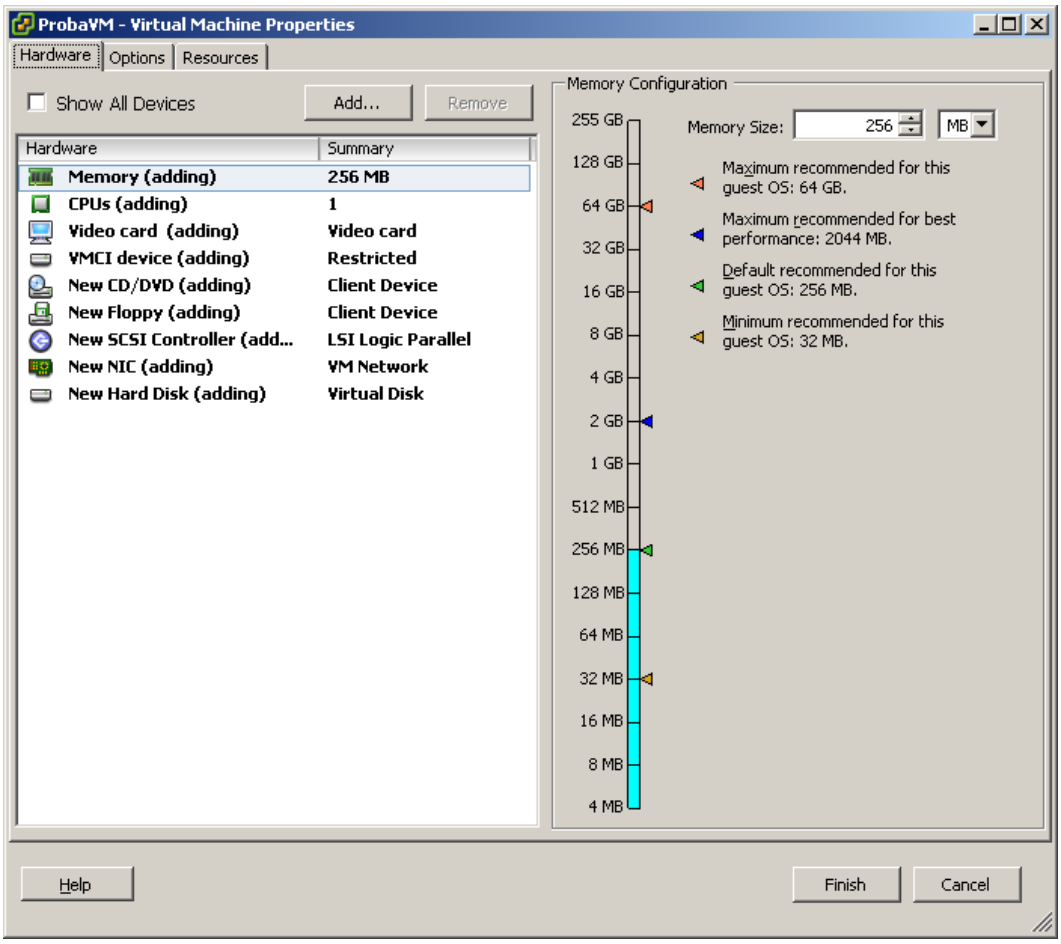
A *vSphere Client* képes letölteni és telepíteni OVF formátumú virtuális gépeket. Az ehhez szükséges funkciók a *File* menüben a *Deploy OVF template...* (már letöltött OVF telepítésére), valamint a *Browse VA marketplace...* (a VMware virtuális készülékeket gyűjtő weboldaláról automatikus letöltés menüpont alatt találhatóak, 20. ábra).

A kiválasztott virtuális készülékből létrejövő virtuális gép legtöbb beállítása az OVF fájl metaadataiból érkezik, néhányat (adattár helye, virtuális gép neve) kézzel kell megadni (21. ábra).

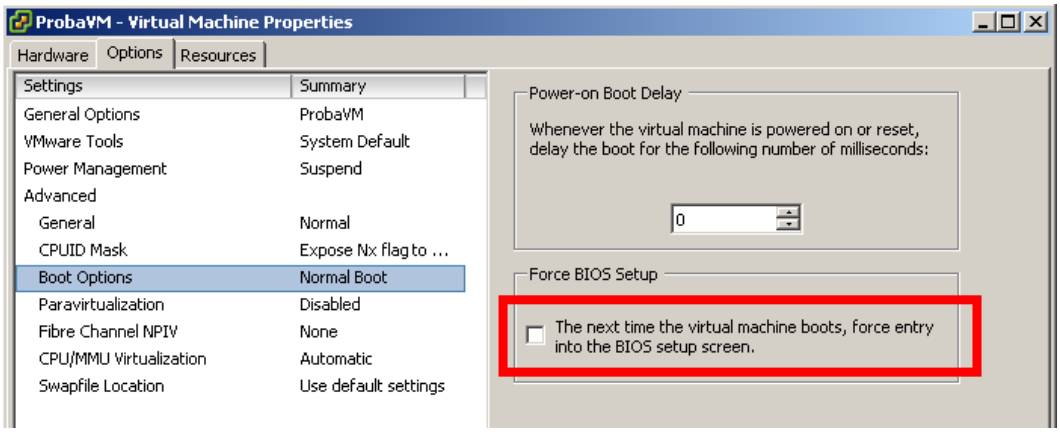
5.2.3. Virtuális gépek feltöltése az adattárba

Az adattár tartalmát úgy tekinthetjük meg, hogy pl a gazdagép Summary fülén a *Datastores* listában a kiválasztott adattár kontextus menüjéből a *Browse datastore...* menüpontot választjuk (22. ábra).

Az adattár böngészőben (23. ábra) tölthetünk fel illetve le fájlokat vagy egész könyvtárakat a klienst futtató gépről az ESXi adattárába. Ahhoz, hogy egy feltöltött virtuális gépet használni tudjunk, be kell regisztrálni a nyilvántartásba (inventory). A virtuális gép könyvtárában a `.vmx` fájl kontextus menüjében található az *Add to inventory...* opció, néhány kérdés után (virtuális gép nevének átírása, helye a nyilvántartásban) a virtuális gép bekerül az erőforrásfába, és indítható lesz. Az erőforrásfából természetesen el is távolíthatóak a virtuális gépek akár a fájlok törlésével



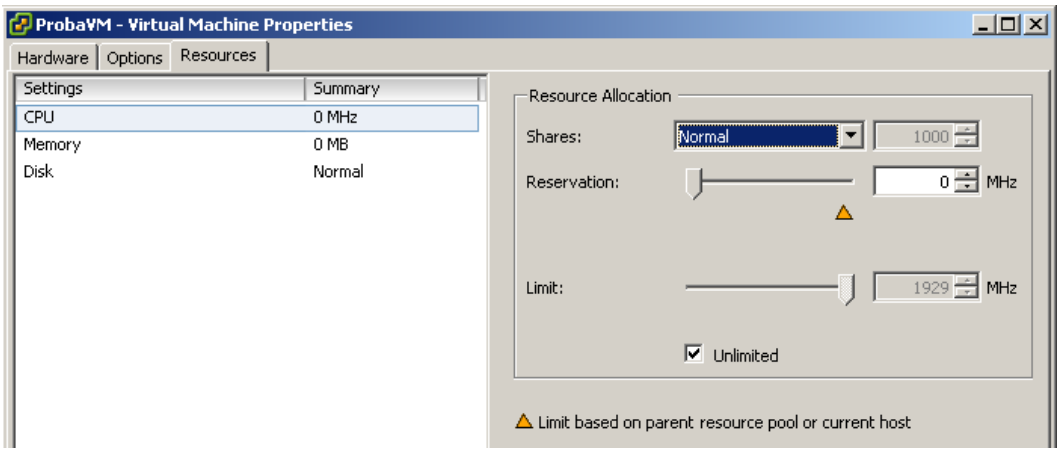
17. ábra. A virtuális gép beállításai: a virtuális hardverelemeket módosító fül



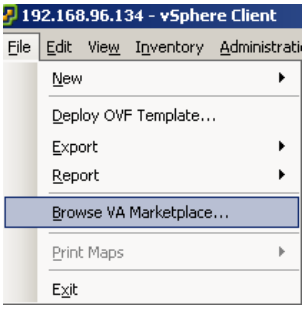
18. ábra. A virtuális gép beállításai: egyéb beállítások (BIOS opciók)

vagy meghagyásával (ez esetben később újra hozzáadható lesz).

Fontos megjegyezni, hogy a Workstationben készített virtuális gépek merevlemez tartalmát tároló vmdk fájlt a feltöltés előtt ESXi kompatibilis formátumra kell konvertálni pl. a `vmware-vdiskmanager` parancssori alkalmazással. Ezen kívül a feltöltés után gyakran a virtuális



19. ábra. A virtuális gép beállításai: erőforrás korlátok és prioritások



20. ábra. A vSphere Client *File* menüje

hálózati adaptert is ki kell cserélni, mert a Workstation eltérően kezeli a virtuális hálózatokhoz rendelést, mint az ESXi.

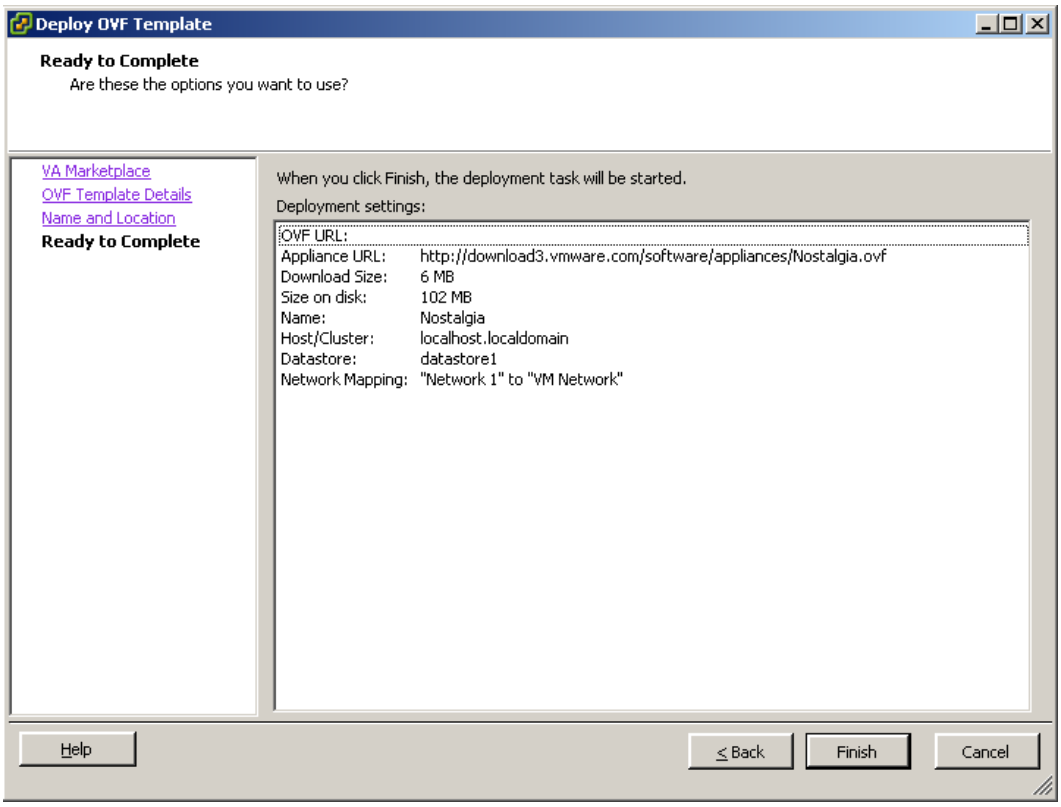
5.3. Virtuális gépek használata

Egy virtualizációs rendszerben a leggyakoribb műveletek a virtuális gépek általános használatához kapcsolódnak:

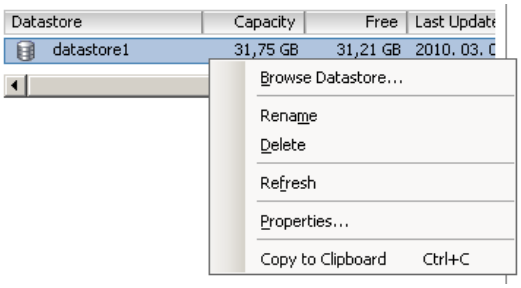
- indítás
- leállítás
- újraindítás
- felfüggesztés (hibernálás, *suspend*)
- konzolhoz kapcsolódás
- konfiguráció módosítás
- állapotmentés (*snapshot*) készítése

Virtuális gépek üzemállapotát a virtuális gép kiválasztásakor a felső sávban megjelenő ikonokkal (play - elindítás, pause - felfüggesztés, stop - leállítás, körbe nyilak - restart) lehet változtatni.

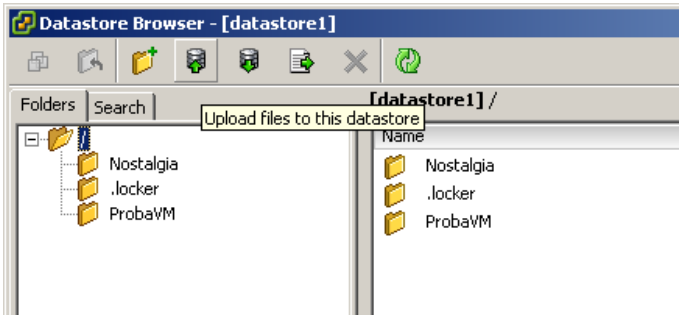
A konzolt a *Console* fülön érhetjük el (24. ábra.), illetve külön ablakba is kirakhatjuk a felső sávban a *Launch virtual machine console* ikonjával. Az ESXi egyazon virtuális géphez tetszőlegesen sok egyidejű kapcsolatot kezel. Több egyidejű kapcsolatnál figyelmeztetést kapunk,



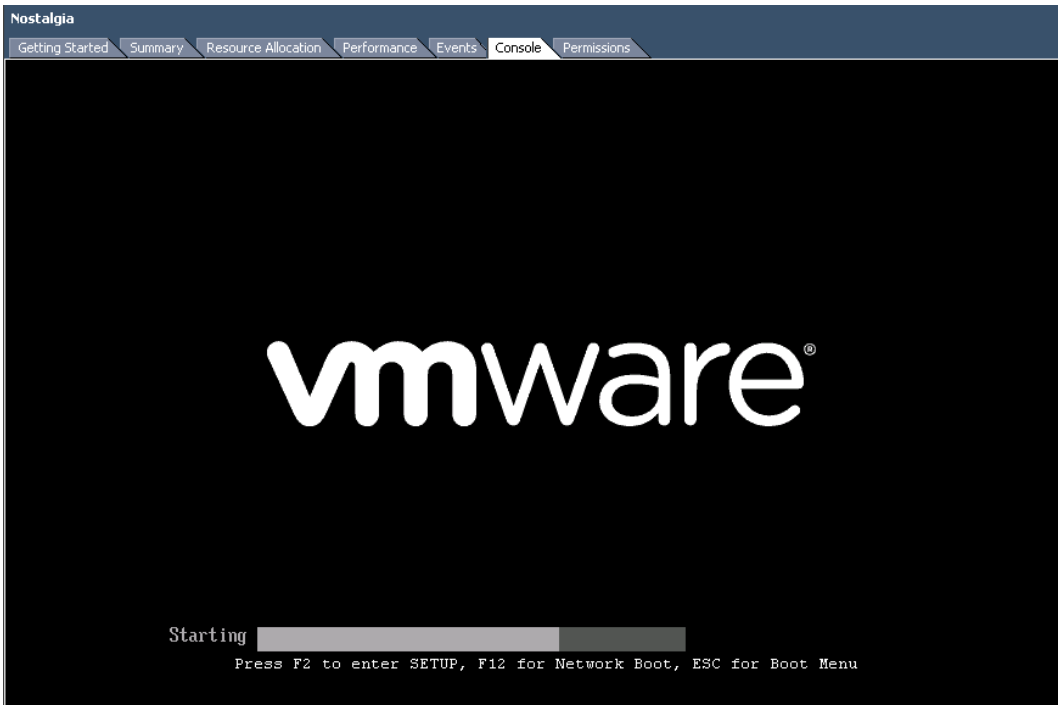
21. ábra. Az előregyártott virtuális gép beállítási lehetőségei



22. ábra. Az adattár kontextus menüje



23. ábra. Adattár böngésző



24. ábra. A virtuális gép konzolját mutató fül

hogy rajtunk kívül más is kapcsolódott a virtuális gép konzoljához. A VMware termékeknél megszokott módon lehet a billentyűzettel és egérrel irányítani a virtuális gép konzolját. A képernyőképbe kattintással fókuszt kap a virtuális gép, innentől kezdve a billentyűzet és az egér a virtuális gépet irányítja. `Ctrl-Alt` lenyomásával lehet kilépni virtuális gépből. A vendéggépbe telepített VMware Tools lehetővé teszi, hogy az egér a virtuális képernyő szélén ki illetve be tudjon lépni virtuális gépbe.

A virtuális gépek *Summary* fülén a fentebb már ismertetett módon módosíthatjuk a virtuális hardver beállításait. A beállítások közül az eltávolítható eszközök (Floppy, CD/DVD, hálózat) a gép futása közben átállíthatóak, a többi beállítás csak leállított vendéggép mellett módosítható. Ebből a szempontból a felfüggesztett állapot is a bekapcsolt állapothoz hasonlít.

Az eltávolítható eszközök azért is érdemelnek külön említést, mert ezeket módosítjuk a leggyakrabban, például ha CD-t akarunk behelyezni a vendéggép virtuális meghajtójába. Az eszközök csatlakoztatott állapotát a *Connected* illetve a *Connect at power on* jelölőnégyzeteivel lehet állítani. Ha bootolni akarunk CD-ről, akkor feltétlenül be kell jelölni a *Connect at power on*-t, mert a BIOS inicializálás gyorsan lefut, ennyi idő alatt kézzel nem lehet csatlakoztatni az eszközt. A virtuális CD lemez lehet egy valódi CD a klienst futtató gép CD meghajtójában (*Client Device*, az ESXi meghajtója *Host Device* vagy egy ISO képfájl (*Datastore ISO*). A leggyakrabban az utóbbit használjuk, a telepítőkészleteket valamilyen adattárra összegyűjtjük, így telepítéskor csak ki kell választani a megfelelőt. A VMware Tools telepítő CD képfájljai is így érhetőek el a `vmimages` könyvtár alatt (ez utóbbi hálózatról indított ESXi esetén nem érhető el).

5.3.1. Állapotmentés

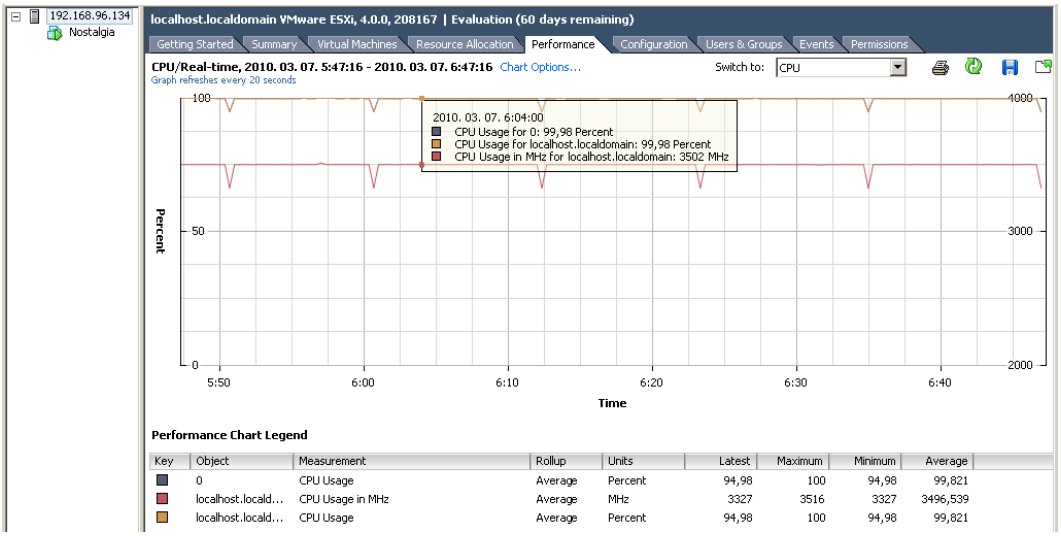
Külön figyelmet érdemel a teljes gépre kiterjedő állapotmentési lehetőség, ami fizikai gépen általában nem megoldható, vagy csak kézi konfigurálással valósítható meg az operációs rendszer szintjén. (Fájlrendszerekről állapotmentés (snapshot) készíthető logikai kötetkezelő (LVM) rendszerekkel, ám a memóriatartalomra ez nem terjed ki.) Az VMware állapotmentő rendszere futó gépről is képes pillanatképet készíteni, így nemcsak a merevlemez tartalma, hanem a memória

pillanatnyi állapota is mentésre kerül. További fontos tulajdonság, hogy egy gépről több, akár egymásból leszármazó vagy közös forrásból kiinduló párhuzamos állapotmentés is kezelhető, és igény szerint bármelyik mentett állapot gyorsan visszatölthető. A *snapshot manager* ikonjai a felső sávon a jobb szélhez közel helyezkednek el.

5.4. Teljesítménymérés és erőforrás-gazdálkodás

A virtualizált rendszerek alaptulajdonsága, hogy közös erőforrásokon osztozó virtualizált környezeteket kezelnek. A virtuális gépek teljesítményét alapvetően meghatározza, hogy a virtualizációs futtatókörnyezet hogyan gazdálkodik a közös erőforrásokkal. Az ESXi számos megfigyelési és beállítási lehetőséget kínál az erőforrások kiosztásának befolyásolására.

Létrehozhatunk csoportosító elemeket (*Resource Group*), melyekbe virtuális gépeket helyezhetünk. A csoportoknak lehet meghatározott erőforrás-korlátja (*Limit*) vagy garantált minimális részesedése (*Reservation*). Korlátot vagy fenntartott minimumot az egyes virtuális gépek szintjén is megadhatunk a korábban már említett *Virtual Machine Properties* ablak *Resources* füle alatt. Így komplex hierarchikus erőforrás-kiosztási fát definiálhatunk, bár leggyakrabban ezt a lehetőséget arra használjuk, hogy az egyes virtuális gépek túlzott erőforrás-foglalását korlátozzuk (pl. egy virtuális gép folyamatosan 100%-ra terhelt CPU-ja ne lassítsa le a többi virtuális gépet). Erőforrások szerinti (CPU, memória, I/O terhelés) prioritásokat is adhatunk a virtuális gépeknek. Ez akkor kap szerepet, ha valamely erőforrásból többre van igényünk, mint amennyi rendelkezésre áll. Ilyenkor a rendszer a prioritások függvényében von el erőforrásokat a vendéggépektől.



25. ábra. Visszamenőleges erőforrás-foglalási és terhelési adatok

Az erőforrás-foglalást illetve a terhelést az ESXi visszamenőleg rögzíti a gazdagépre valamint minden virtuális gépre. A vCenter Server ezt kiterjeszti saját adatbázissal, ami akár 1 évre visszamenő historikus adatokat is tárolhat. A visszamenőleges adatok grafikonok formájában megtekinthetőek a *Performance* fül alatt (25. ábra). A grafikonokon megjeleníthető értékek és az időbeli felbontás a *Change Chart Options...* alatt választhatóak ki.

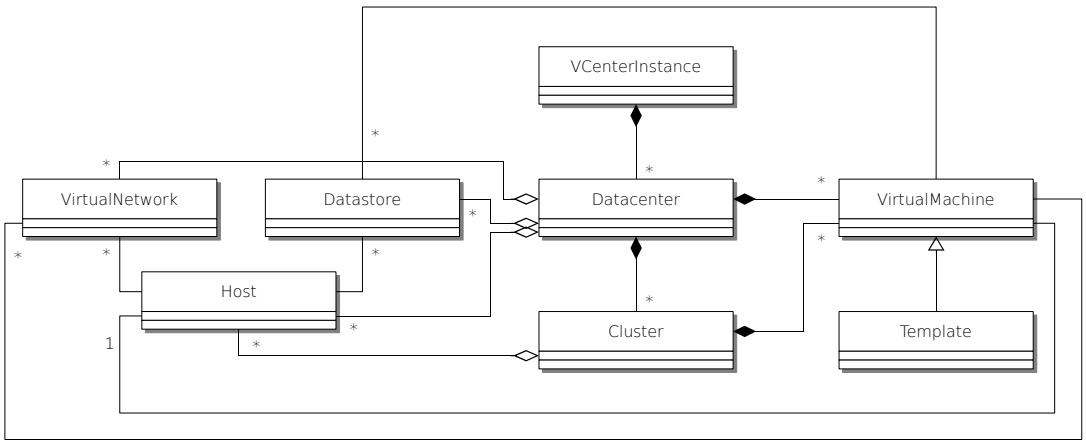
A virtuális gépen belüli óra és a gazdagép valós idejű órája sok esetben nem szinkronizált (óraszinkron szolgáltatást a VMware Tools biztosít), jelentős elcsúszás lehet közöttük, ami ráadásul nem feltétlenül konstans, függhet a terheléstől[6]. Emiatt a virtuális gépen belüli teljesítményméréseknél nem szabad kizárólag a belső órára hagyatkozni.

[6]Az ESXi mostani kiadásában az óraelcsúszási probléma csak akkor jelentkezik, ha a gazdagép CPU-ja nem állandó órajelen megy, hanem energiatakarékossági célból kis terhelésnél lelassul.

6. Központi felügyelet vCenter Serverrel

A vCenter Server (régebbi nevén Virtual Center) számos ESXi gazdagép egyidejű központi felügyeletét végzi, amit az alábbi új funkciókkal egészít ki:

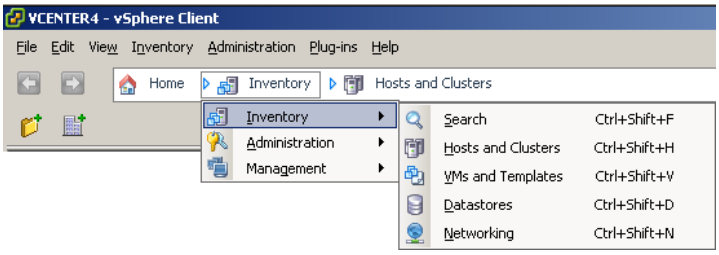
- Központosított **jogosultságkezelés**, virtuális gépeken végezhető műveletek szabályozott delegálása felhasználóknak
- Virtuális gép **sablonok** kezelése, sablon alapján előtelepített operációs rendszerrel rendelkező gépek példányosítása
- Gazdagépek fürtbe szervezése
  - **Működés közbeni áthelyezés** (live migration, *VMotion*)
  - Automatikus **terheléselosztás** gazdagépek között (*Distributed Resource Scheduling*, *VMware DRS*)
  - **Hibatűrés**, gazdagép kiesés automatikus kezelése (*High Availability*, *VMware HA* illetve *Fault Tolerance*, *VMware FT*)



26. ábra. vCenter fogalmai és kapcsolatai (nem teljes metamodel)

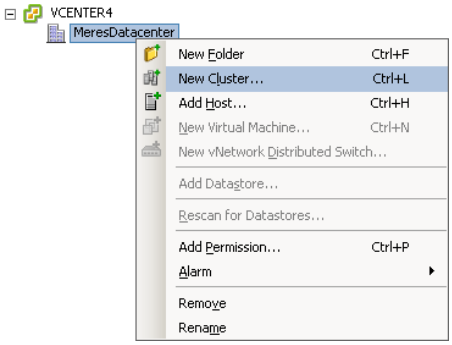
6.1. A vCenter kezelése

A vCenter Serverhez szintén a vSphere Client szoftverrel kapcsolódhatunk, az ESXi és a vCenter közös web service alapú API-t használ. Kapcsolódás után a kezelőfelületen elérhető funkciók kibővülnek.



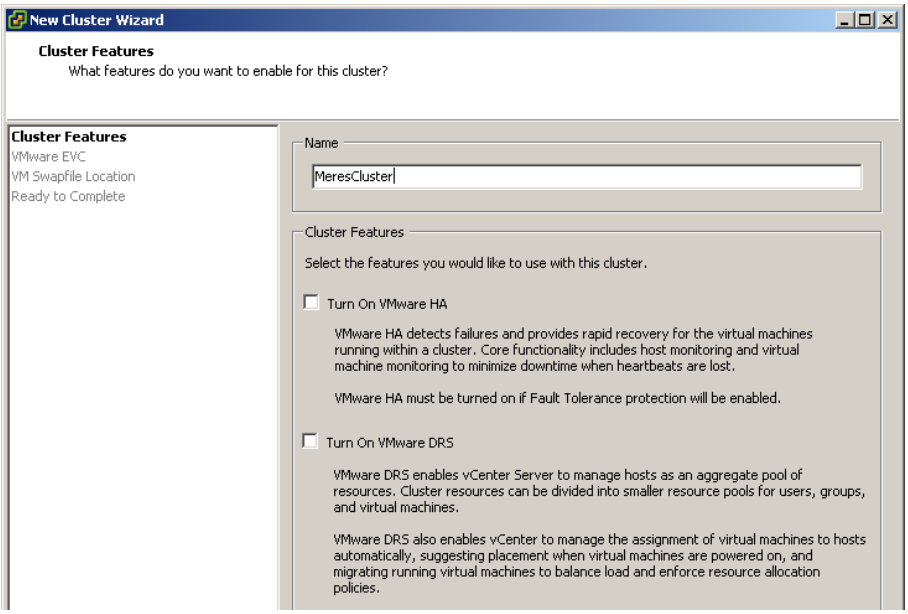
27. ábra. A vCenter Server új nézetei

A legfontosabb eltérés, hogy új nézetek jelennek meg (27. ábra). Bizonyos típusú konfigurációs elemek (gazdagépek, hálózatok, adattárak, sablonok) csak egy-egy nézetben, mások több nézetben is látszanak (virtuális gépek, adatközpontok, fürtök).



28. ábra. Fürtök a vCenter Serverben

A nyilvántartásban megjelenő konfigurációs elemek típusai is bővülnek (26. ábra), megjelenik fő tartalmazó elemként az adatközpont (*datacenter*) valamint az adatközponton belül a fürt (*claster*) fogalom (28. ábra). Lehetnek a nyilvántartásban virtuális gép sablonok is bejegyezve.



29. ábra. A vCenter Server terheléselosztási és hibatűrési funkciói

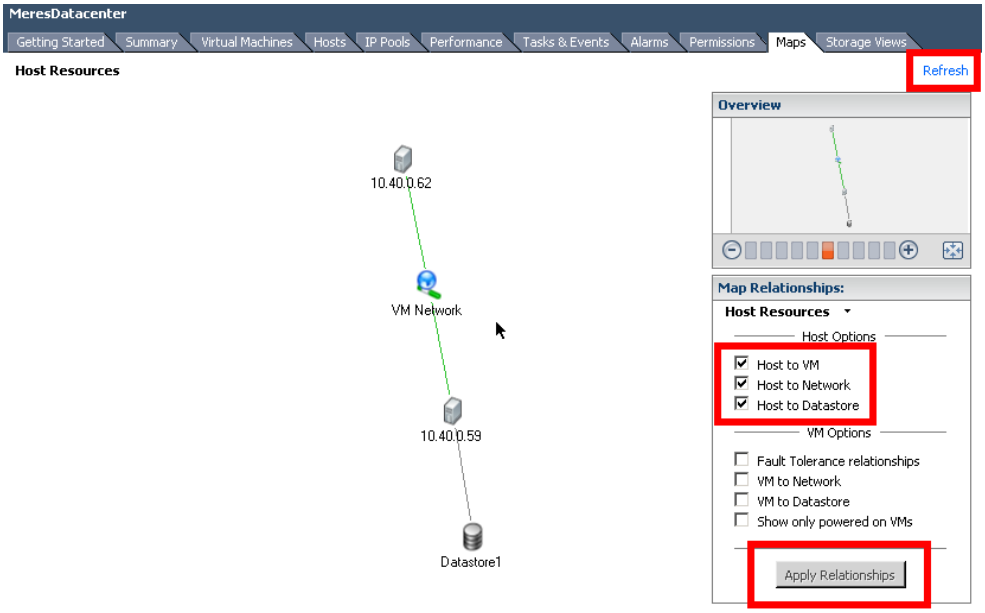
Fürt létrehozásakor illetve később a cluster kontextus menüjéből az *Edit Settings...* menőpontot kiválasztva bekapcsolhatjuk a terheléselosztás és a hibatűrési funkciót (29. ábra).

Egy gazdagépet elhelyezhetünk egy adatközpontban vagy egy fürtben is az *Add Host...* menüponttal. Ilyenkor az ESXi-n a root felhasználónevet és a beállított jelszót kell megadni, valamint meg kell adni egy helyet a nyilvántartásban (inventory), ahova a gazdagépen esetlegesen talált virtuális gépek kerülni fognak. A gazdagépeket a későbbiekben át is helyezhetjük a nyilvántartáson belül, például egy fürtből kivehetjük a adatközpontba, ám már futó virtuális gépek esetén ez bonyolult átkonfigurálást is jelenthet.

A felvett gazdagépeken létező hálózatok és adattárak is automatikusan megjelennek a nyilvántartásban. Természetesen új adattárat és hálózatot is definiálhatunk pontosan úgy, ahogyan a különálló ESXi esetén tettük: a megfelelő gazdagépet ki kell választani, és a *Configuration*

fül alatt elvégezhetünk minden gazdagép-specifikus beállítást. Fontos megjegyezni, hogy a vCenter az adattárak esetén automatikusan képes (UUID alapján) felismerni a több gazdagépről látható közös adattárakat. Ellenben a hálózatok esetén nincsen ilyen lehetőség, így a vCenter az egyes virtuális hálózatok neveire hagyatkozik az azonos hálózatok beazonosításakor. Tehát fontos, hogy az azonos fizikai hálózatra kivezetett virtuális hálózatok minden gazdagépen pontosan ugyanazt a nevet kapják, eltérő hálózatok viszont ne kapjanak azonos nevet. Az egységes elnevezések a virtuális gépek szempontjából is fontosak, mert itt is név alapján történik a hálózati kapcsolatok hozzárendelése, amikor egy virtuális gépet más gazdagépre helyezünk át.

A gazdagépek, adattárak és hálózatok valamint későbbiekben a virtuális gépek összerendeléseit a térkép a (*Maps*) nézeten grafikusan is megtekinthetjük. Három különböző látókörű térkép nézet van: egy globális, egy adatközponthoz kötött és egy virtuális géphez kötött. A globális nézet a navigátor sávban a `Home > Management > Maps` menüpont alatt látható, ilyenkor külön-külön jelölőnégyzettel választhatjuk ki, hogy a nyilvántartásból mely objektumok jelenjenek meg. Ha a navigátor sávban a `Home > Inventory` valamelyik nézete aktív, és a nyilvántartásból kiválasztunk egy Datacentert vagy Clustert, akkor a Maps *fülön* láthatjuk az adott adatközpont/fürt objektumait. Ha virtuális gépet választunk ki, akkor szintén a Maps *fülön* láthatjuk az adott virtuális géphez kapcsolódó erőforrásokat.



30. ábra. Az objektumok közötti kapcsolatok típus szerinti szűrése (alapértelmezetten nem minden kapcsolat látható)

Az első két nézetben a térkép jobb szélén találjuk azokat a beállításokat, amelyekkel az objektumok közötti kapcsolatokat típusuk szerint szűrhetjük. Erre érdemes odafigyelni, mert alapértelmezetten nem minden kapcsolat látható (30. ábra). Változások esetén a térkép frissítése manuálisan történik a jobb felső sarokban található *Refresh* gombbal.

6.2. Működés közbeni áthelyezés VMotionnel

A virtuális gépek működés közbeni áthelyezése (live migration) azt jelenti, hogy a vendéggép teljes futási állapota az egyik gazdagépről egy másikra kerül át anélkül, hogy a vendéggépet le kellene állítani, vagy akár csak néhány másodpercnél hosszabb időre fel kelljen függeszteni. Ennek elsődleges célja, hogy a gazdagépek *tervezett* leállásai (hardver karbantartás, szoftver frissítés telepítése) esetén se kelljen a tényleges szolgáltatásokat nyújtó virtuális gépek működését szüneteltetni. Nyilvánvaló, hogy ez a technológia nem nyújt védelmet a *nem tervezett* leállásokkal

szemben.

Megosztott háttértáron tárolt virtuális merevlemezek esetén csak a vendéggép memóriatartalmát és a CPU állapotát kell áthelyezni. Könnyen belátható, hogy ez még kis memóriaméret esetén sem atomi művelet, néhány GB-os memóriaméret esetén pedig még gigabites áteresztőképességű hálózaton is perc nagyságrendű időbe kerülhet. Ez túl hosszú idő, a vendéggép külső hálózati kapcsolatai ennyi idő alatt lebontódnak, ami észrevehető kiesést okozhat a szolgáltatásokban.

A fentiek miatt a legtöbb működés közbeni áthelyezést biztosító megoldás halasztott (deferred) áthelyezést használ: a memóriatartalom másolását még azelőtt megkezdi, hogy az első gazdagépen a vendéggép futása leállna, és csak a közel teljes másolat elkészültekor viszi át a CPU állapotát, és adja át a vezérlést a második gazdagépen futó másolat számára. Problémát okoz, hogy a másolás közben már átvitt memórialapokban az első gépen még futó vendéggép képes módosításokat végezni, így érvénytelenné teheti a másolat egyes részeit. Ezeket a módosult memórialapokat az első másolási fázisban külön meg kell jelölni, és a vezérlésátadás után egy második fázisban át kell vinni. A második másolási fázis alatt újfent működik a vendéggép, de már a második gazdagépen, ilyenkor az okozhat problémát, ha olyan memórialapot akar olvasni, ami eredetileg még az első gazdagépen történt módosítás miatt érvénytelenné vált, és még nem érkezett meg az érvényes másolata a második gépre. A virtualizált környezetben azonban a vendég operációs rendszer tudta nélkül úgy állíthatóak a memórialap-táblák, hogy a CPU érvénytelen memóriahozzáférési kivételt generáljon, ha ilyen érvénytelennek jelölt lapokhoz történne hozzáférés. Ilyenkor – az általános elfogás és emulációs elvnek megfelelően – a vezérlés a hypervisorhoz kerül, ami a kivételt kezeli. Soron kívül áttölti az első gazdagépről a kért memórialapot, majd visszaadja a vezérlést a virtuális gépnek, ami folytatja a futását, és sikeresen hozzáfér a kért memórialaphoz. Ez a működési elv nagyban hasonlít – az operációs rendszerekben is alkalmazott – merevlemezre történő lapozáshoz (swap), ennek megfelelően jelentős teljesítményvesztéssel is jár a használata, hiszen a virtuális gép futásának addig szünetelnie kell, amíg a memórialap meg nem érkezik a hálózaton át.

Könnyen belátható, hogy a virtuális gépen belüli aktivitás, az éppen ténylegesen aktívan módosított memóriaterület nagysága befolyásolja, hogy az áthelyezés mennyire gyorsan történik, valamint hogy a vezérlésátadáskor a kiesés mennyire lesz hosszú, a kiesés után mennyi idő még, amíg a lap áttöltések miatt teljesítménycsökkenéssel kell számolni.

A VMware VMotion technológiájának használata esetén fontos kijelölni, hogy mely hálózati interfészt akarjuk használni a gazdagépek közötti memóriatartalom átadásához. A memóriatartalom másolását a kernel végzi el, tehát a kernel hálózati interfészén kell a beállítást elvégezni.

6.3. Automatikus terheléselosztás

A működés közbeni áthelyezés egy másik alkalmazása, amikor a vendéggépeket a terhelés függvényében újraoszthatjuk a futtató gazdagépek között. Például ha az egy fürtben található gazdagépek egyikén magas a CPU-használat, viszont van a fürtben nagyon alacsonyan kihasznált CPU-val rendelkező gazdagép is, akkor érdemes lehet néhány virtuális gépet áthelyezni a terhelt gazdagépről a terheletlenre, így az összteljesítmény és a rendelkezésre álló hardverek kihasználtsága is javul. Hasonlóképpen az erősen egyenetlen memóriahasználat esetén is érdemes lehet áthelyezést végezni.

Az újraelosztás vezérlését egy központi döntéshozó komponens végzi, ami a jelen esetben a vCenter Server részeként fut, monitorozza az egyes gazdagépek terhelését valamint a terhelést generáló vendéggépeket. Ha huzamosabb időn keresztül átlagolva is egyenetlen az eloszlás (a gazdagépek terhelésének szórásnégyzete meghalad egy küszöbértéket), akkor egy optimumkereső algoritmus segítségével kiválaszt egy-egy áthelyezendő vendéggépet. A döntési algoritmus számos paramétere finomhangolható, a két legfontosabb a szórásnégyzet küszöbértéke és az átlagolási idő. Ez azért fontos, mert az áthelyezés költséges, maga is terhelést generál, valamint rontja az éppen áthelyezett gép teljesítményét, ezért például rövid tranziens terhelések esetén a rendszeres ide-oda áthelyezések hatása a teljesítményre káros is lehet.