



Budapesti Műszaki és Gazdaságtudományi Egyetem

Villamosmérnöki és Informatikai Kar

Irányítástechnika és Informatika Tanszék

Kiszolgálók üzemeltetése nagyvállalati környezetben és kapcsolódó szoftverek fejlesztése

SZAKDOLGOZAT

Készítette

Főglein Simon István

Konzulens

dr. Somogyi Péter

2024. május 2.

Tartalomjegyzék

Tartalomjegyzék

Kivonat	i
Abstract	ii
1. Bevezetés	1
1.1. Nagyvállalati környezetek ismertetése	3
1.2. Virtualizáció	5
1.2.1. Népszerű virtualizációs technológiák	6
1.2.2. Virtuális gépek használatának néhány előnye	7
1.2.2.1. Erőforrások testreszabása	7
1.2.2.2. Snapshotok	8
1.2.2.3. Migráció	8
1.2.3. Teljes virtualizáció és paravirtualizáció összehasonlítása	9
1.2.4. Konténerizáció	9
1.3. Logikai kötetkezelés	9
1.3.1. Snapshotok, mentések készítése	9
2. Technológiai áttekintés	10
2.1. Szervergépek	10
2.2. Virtualizáció	11
2.2.1. Áttekintés	11
2.2.2. Paravirtualizáció és teljes virtualizáció	12
2.2.3. Virtualizációs lehetőségek összehasonlítása	12
2.3. Logikai kötetkezelés	13
2.4. OS-lehetőségek	15
2.5. Eszközmenedzsment	15

2.5.1. Ansible és Salt	15
2.6. Monitoring	15
3. Saját munka bemutatása	16
3.1. Kialakítani kívánt környezet meghatározása	16
3.2. Fizikai gép ismertetése	17
3.3. Operációs rendszer	17
3.3.1. Debian, Ubuntu	17
3.3.2. SUSE	17
3.4. Gépmenedzsment: Salt	17
3.4.1. Más hasonló megoldások	17
3.5. Monitoring	17
3.6. Továbbfejlesztési lehetőségek	17
Köszönhetetlenül	18
Szójegyzék	19
Acronyms	20
Irodalomjegyzék	20
Irodalomjegyzék	21

HALLGATÓI NYILATKOZAT

Alulírott *Főglein Simon István*, szigorló hallgató kijelentem, hogy ezt a szakdolgozatot meg nem engedett segítség nélkül, saját magam készítettem, csak a megadott forrásokat (szakirodalom, eszközök stb.) használtam fel. minden olyan részt, melyet szó szerint, vagy azonos értelemben, de átfogalmazva más forrásból átvettem, egyértelműen, a forrás megadásával megjelöltem.

Hozzájárulok, hogy a jelen munkám alapadatait (szerző(k), cím, angol és magyar nyelvű tartalmi kivonat, készítés éve, konzulens(ek) neve) a BME VIK nyilvánosan hozzáférhető elektronikus formában, a munka teljes szövegét pedig az egyetem belső hálózatán keresztül (vagy autentikált felhasználók számára) közzétegye. Kijelentem, hogy a benyújtott munka és annak elektronikus verziója megegyezik. Dékáni engedéllyel titkosított diplomatervezet esetén a dolgozat szövege csak 3 év eltelte után válik hozzáférhetővé.

Budapest, 2024. május 2.

Főglein Simon István

hallgató

Kivonat

Jelen dokumentum egy diplomaterv sablon, amely formai keretet ad a BME Villamos-mérnöki és Informatikai Karán végző hallgatók által elkészítendő szakdolgozatnak és diplomatervnek. A sablon használata opcionális. Ez a sablon \LaTeX alapú, a \TeX Live \TeX -implementációval és a PDF- \LaTeX fordítóval működőképes.

Abstract

This document is a L^AT_EX-based skeleton for BSc/MSc theses of students at the Electrical Engineering and Informatics Faculty, Budapest University of Technology and Economics. The usage of this skeleton is optional. It has been tested with the *TeXLive* T_EX implementation, and it requires the PDF-L^AT_EX compiler.

1. fejezet

Bevezetés

Napjainkban az informatika és az internet életünk szerves részévé vált. A számos szolgáltatás folyamatos rendelkezésre állásának biztosítása és a megnövekedett forgalom kiszolgálása jó néhány új technológia kifejlesztését követelte meg.

Dolgozatomban elsősorban egy általános képet szeretnék adni arról, hogy milyen üzemeltetési kihívásokkal kell szembenéznünk, ha egy ilyen szolgáltatás működtetésébe vágjuk a fejszénket. Értekezésemben nem fogok kitérni bizonyos infrastrukturális hátterekre – mint például a kiszolgálók folyamatos energiaellátásának biztosítása –, ezeket adottnak fogom tekinteni, hiszen ezt egy adatközpontban bérelt hely esetén sem magunknak kell biztosítanunk. A következőkben sokkal inkább az informatikai lehetőségek tárgyalására fogom helyezni a hangsúlyt: hogyan tudunk hatékonyan üzemeltetni több kiszolgálót, milyen módon lehet biztosítani a szolgáltatásaink lehető legnagyobb rendelkezésre állását, és hogyan védhetjük meg adatainkat egy esetlegesen félresikerült rendszerfrissítést követően.

A dolgozatban érinteni fogom a jelenleg legelterjedtebb virtualizációs technológiákat, melyek főbb tulajdonságait röviden ismertetem, valamint össze is hasonlítom ezeket a megoldásokat a legfontosabb különbségekre kitérve.

Szerepet fog kapni továbbá a logikai kötetkezelés, ezen belül is a Linux kernelben elérhető LVM-implementáció. Ez a technológia nagyban megkönnyíti a háttértárak és partíciók kezelését üzemeltetési szempontból, melyet főként virtualizációt végző fizikai gépek esetében használhatunk ki, hiszen ilyen helyzetekben érdemes minden virtuális rendszernek külön partíciót létrehozni, amelyek kezelése (pl. egy esetleges bővítés során) a hagyományos particionálási megoldásokkal sokkal összetettebb feladat lenne.

Szót ejtek a monitoring megoldásokról is, melyek elengedhetetlenek ahhoz, hogy a rendszer üzemeltetését végző szakemberek pontos képet kapjanak az infrastruktúra ak-

tuális állapotáról, az esetleges korábbi problémákról. A monitorozás azért is fontos, mert ha egy hibát ezáltal sikerül idejekorán felismerni (például háttértáراك esetén egy megfelelő határék beállításával időben értesülhetünk egy partíció megteléséről, és nem csak az írási hibákat tapasztaljuk), akkor elkerülhetők a további, komolyabb hibák, amik akár a felhasználók számára is fennakadásokat okozhatnak. Az általam létrehozott teszkörnyezetben is bemutatok egy ilyen monitoring megoldást, melynek segítségével az általam létrehozott infrastruktúra gépeit fogom folyamatosan ellenőrizni.

A teszkörnyezet beállításában nagy szerep fog jutni a választott konfigurációmenedzsment szoftvernek, a Salt-nak. Ez arra fog lehetőséget biztosítani, hogy egyes konfigurációs fájlokat egyszerűen telepíthessünk több számítógépre is, valamint a keretrendszer leírónyelvén meghatározott konfigurációleíró szoftver lehetővé teszi azt is, hogy ellenőrizzük egyes szolgáltatások (service) állapotát. Ez hasznunkra válhat például egy saját serviceszel érkező program telepítését követően, hiszen így a leíróban megadhatjuk a telepítés paramétereit, majd ezt követően egyből ellenőrizhetjük is, hogy a telepítés után sikeresen elindult-e az újonnan telepített szoftver.

A dolgozatban tárgyalt koncepciókat egy kisebb volumenű tesztrendszeren keresztül fogom bemutatni. Ennek a rendszernek a célja nem egy teljes vállalati környezet bemutatása, hiszen ehhez nagy mennyiségű hardverre, jelentős mértékű hardveres és szoftveres erőforrásokra lenne szükség, amelyek üzembe helyezése, összehangolása túlmutat a dolgozat keretein. Ehelyett sokkal inkább arra szeretném rávilágítani, hogy milyen eszközök állnak rendelkezésre egy ilyen nagyszabású infrastruktúra sikeres üzemeltetésének elősegítéséhez. Gondoljunk csak arra, hogy egy 5-10 számítógépből álló rendszer esetén kivitelzhető, hogy a rendszergazdák egyesével telepítsék a havi frissítéseket, azonban egy több száz, vagy több ezer kiszolgálóból álló nagyvállalati környezetben nem lenne egy realis elvárás.

Az ilyen és ehhez hasonló kihívások megoldására fogok lehetőségeket mutatni a 2. fejezetben. Szó lesz a gépek távoli kezeléséről, folyamatos karbantartásukról, automatikus biztonsági javításokról (patchek) való értesülésről, ezek telepítéséről. Tárgyalni fogom továbbá a rendszert alkotó eszközök monitorozását, metrikák gyűjtését is, továbbá szó lesz az egyre szélesebb körben elterjedő konténerizációs technológiákról, ezek használatáról vállalati környezetekben. Bemutatom azt is, hogy a megfelelő eszközökkel milyen gyorsan hozhatunk létre konténereket, és mennyire hatékonyan kezelhetjük őket akár egy böngészőből is. Fontos megjegyezni, hogy az itt említett technológiák kisebb környezetekben is használhatóak, azonban néhány esetben az ilyen rendszerek használata kevesebb előnyt

nyújt, mint amennyi munkát telepítésük és karbantartásuk igényel, így érdemes felmérni az informatikai rendszerrel szemben támasztott elvárásainkat, és ennek megfelelően dönteni a szükséges technológiai komponensekről.

A 3. fejezetben fogom ismertetni az általam készített tesztkörnyezetet, ennek felépítését, a tervezési döntéseket, komponenseit, valamint az ezzel kapcsolatos munkáim során felmerült nehézségeket, tapasztalatokat. Ebben a fejezetben a korábban tárgyalt technológiák közül általam választott megoldásokat

1.1. Nagyvállalati környezetek ismertetése

A vállalatok egyre nagyobb hangsúlyt fektetnek az informatikai rendszereik fejlesztésére és karbantartására, üzemeltetésére. Az ezek által nyújtott szolgáltatások sok esetben jelentős könnyebbéget jelentenek egyes üzleti folyamatokban, és a megfelelően automatizált munkafolyamatok csökkentik az egyes munkavállalók által elvégzendő manuális feladatokat. Fontos tisztában lenni azonban azzal, hogy ezek a megoldások csak akkor működnek jól a minden nap használat során, ha sikerül biztosítani a megfelelő rendelkezésre állást, tehát egy – a vállalat munkavállalói által a munkához nélkülözhetetlen – szolgáltatásnak munkaidőben folyamatosan elérhetőnek kell lennie. Előfordulhatnak olyan igények is, amik miatt bizonyos, a vállalat működésének szempontjából nélkülözhetetlen szolgáltatásoknak folyamatosan elérhetőnek kell lenniük, mert a működésük nem munkaidőhöz kötött (ilyen lehet például a szervezet weboldala, illetve levelezőszervere). Emellett a fent említett informatikai rendszerek karbantartásának is sokszor észrevehetetlennek kell lennie, azaz egy esetleges frissítés nem hátráltathatja a munkavégzést és a rendelkezésre állást. Az ilyen helyzetek kezelésére számos megoldás jött létre, ezek közül néhányat ismertetni fogok a következő fejezetben.

Egy másik fontos szempont a nagyvállalati rendszerek üzemeltetése – és általában informatikai megoldások üzemeltetése és használata során – ami napjainkban egyre nagyobb figyelmet kap, az IT-biztonság kérdése. A rendszerüzemeltetőknek tisztában kell lenniük a potenciális veszélyekkel, veszélyforrásokkal és fel kell készülniük egy esetleges támadásra, annak kezelésére. Sokszor hallhatunk a rendszeres biztonsági mentések fontosságáról, és ezek típusairól, követelményeiről. Egy jól bevett gyakorlat például az úgynevezett 3-2-1 mentési stratégia, ami egy jó kiindulási alapul szolgálhat minden szervezet számára a biztonsági mentésekhez. [8] A megoldás neve az alábbi elvekből származik: [3]

- három példány az adatokról,

- két különböző eszközön (akár más típusú adathordozókon, pl. SSD, HDD, mágnesszalag – ez segít az adathordozóra jellemző esetleges hibák hatásának csökkentésében),
- egy példányt földrajzilag különböző helyen tároljunk (pl. a cég székhelyén legyenek az eredeti adatok és még egy mentés, és egy példányt pedig tároljunk adatközpontban, vagy vegyük igénybe harmadik féltől biztonságimentés-szolgáltatást).

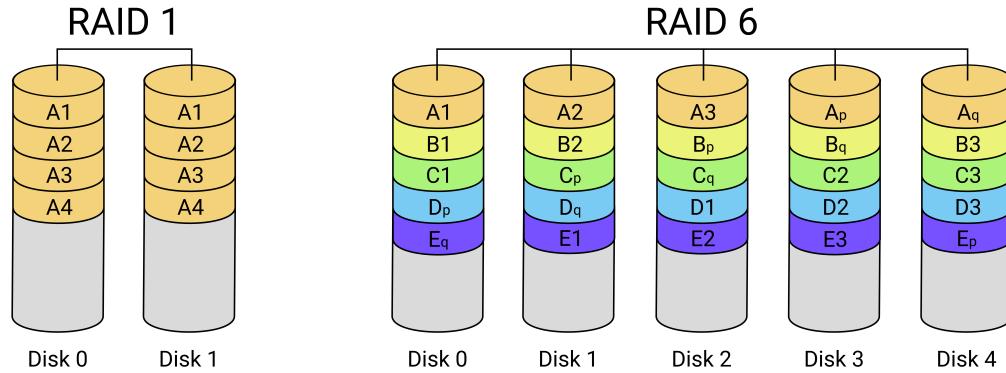
A biztonsági mentések elvégzésére és automatizálására többféle megoldást választhat a szervezet az igényeihez igazodva. Bevett szokás például, hogy a cégen belüli mentéseket valamilyen mentést támogató vagy akár teljesen automatizáló szoftverrel oldják meg (például Bareos, Bacula, BackupPC). Az ilyen megoldások üzembe helyezése nehezebb lehet, mintha például csak egyéni scriptekkel hajtanánk végre a mentéseket, de hosszabb távon mégis célszerű lehet megfelelően konfigurálni őket, mert könnyebben kezelhetővé teszik egy komplex infrastruktúrában található gépek adatainak mentését, illetve az egyszer megírt konfigurációs fájlok több gépen is felhasználhatóak. Mindezek mellett ezek a szoftverek általában rendelkeznek valamilyen grafikus felhasználói felülettel (pl. webes interfész), amely tovább egyszerűsíti a mentések készítését, valamint szükség esetén a mentés visszaállítását.

The screenshot shows the Bareos WebUI interface. At the top, there's a navigation bar with links for Dashboard, Jobs, Restore, Clients, Schedules, Storages, and Director. Below the navigation is a summary section titled 'Jobs started during the past 24 hours' with five categories: Running (3), Waiting (3), Successful (55), Warning (1), and Failed (1). To the right of this is a 'Job Totals' table showing 64,995 jobs, 428,858 files, and 75.88 TB of bytes. Below these are two main sections: 'Running jobs' and 'Most recent job status per job name'. The 'Running jobs' section lists several jobs with their status (e.g., Success, Warning, Failed) and details like Job ID, Client, Level, Start, End, and Bytes. The 'Most recent job status per job name' section lists jobs grouped by name, showing the latest status for each. At the bottom, there are pagination controls for the tables.

1.1. ábra. A Bareos biztonsági mentéseket végző program webes felületének részlete. Forrás: <https://www.bareos.com/bareos-webui-modules/>

Nagyvállalati környezetben nem hagyhatjuk ki a RAID-megoldásokat (a népszerű RAID 1 és RAID 6 megoldásokat az 1.2 ábra szemlélteti), ha szerverek biztonsági mentéséről beszélünk. Ezek arra adnak lehetőséget, hogy az adatokat több háttértáron (pl. merevlemez vagy SSD) tároljuk úgy, hogy egy esetleges diszk hiba ne okozzon fennakadást a működésben. Fontos tisztában lenni azonban azzal, hogy a RAID-megoldások nem védenek bizonyos veszélyek ellen (például zsarolóból vírusok), hiszen az adatok duplikálása valós időben történik, így egy esetleges támadás során a RAID poolba bevont összes disz-

ken megváltoznak az adatok, így nem alkalmas a támadás utáni visszaállításra. Emiatt egy RAID pool a 3-2-1 mentési stratégiát alkalmazva csak egyetlen eszköznek tekinthető, hiába több lemezt használunk a mentés során. RAID-elést tehát csak hardveres hibák ellen érdemes használnunk, rosszindulatú támadás esetén ezek nem nyújtanak védelmet az adataink számára.



1.2. ábra. RAID 1 és RAID 6 megoldások felépítése [6]

A legtöbb hétköznapi felhasználó számára ismeretlen vagy meglepő lehet, hogy maga az internet és az ezen keresztül elérhető szolgáltatások – gondoljunk például az Ügyfélkappa vagy az internetbank-szolgáltatásokra – nagyon komplex rendszerek nem csak szoftveres, hanem informatikai infrastruktúra szempontjából is. A legtöbb ilyen szolgáltatás egy adatközpontban lévő szerveren fut, ami a beérkező kérésekre ad válaszokat. Ezt a folyamatot úgy is felfoghatjuk, hogy az ilyen szolgáltatások felhasználói lényegében az adott szolgáltató (a fenti példánál maradva a Magyar Állam és az adott bankok) számítógépeivel kommunikálnak.

Ezek a szervergépek több lényeges különbséggel is bírnak a személyi számítógépekkel szemben. Egyik legfontosabb tulajdonságuk, hogy hibatűrők bizonyos hardverhibákat illetően: szinte minden főbb komponensből legalább kettő áll rendelkezésre, így ha az egyik meg is hibásodik, akkor a hiba elhárításáig a beépített redundancia miatt a gép képes tovább funkcionálni, általában a felhasználók felé észrevétlenül, míg a gép üzemeltetői figyelmeztetést kapnak a hiba típusáról és a kapcsolódó tennivalókról.

1.2. Virtualizáció

A fent említett megnövekedett forgalom kiszolgálását hatékonyan lehet kezelní úgy, hogy olyan fizikai számítógépet helyezünk üzemmbe, mely több, egymástól független operációs rendszer futtatására is alkalmas. Ilyenkor ezeket a fizikai gépen futó rendszereket virtuális

gépeknél (virtual machine, VM) nevezzük. Egy virtuális gép elkölöntött erőforrásokat kap a fizikai géptöl, hozzáférhet például bizonyos mennyiségek processzormaghoz, memóriához, illetve külön háttértár-partíciói is lehetnek. A virtualizált hardverek és operációs rendszerek a legtöbb esetben a külvilág felé nem különböztethetőek meg a fizikai számítógépektöl, és ezzel a megoldással jelentősen csökkenthető a rendszerek és a hozzájuk szükséges informatikai infrastruktúra üzemeltetésének költsége.

A virtualizáció nagy ereje abban rejlik, hogy bizonyos hardverek virtualizációjával egységesítményt olcsóbban kaphatunk meg, mintha külön fizikai gépeket helyeznének üzembe, illetve nagyobb rugalmasságot kapunk a kezelésükben, üzemeltetésükben. Képzeljük el, hogy megveszünk egy számítógépet, amin szeretnénk futtatni egy számunkra fontos alkalmazást, mondjuk a honlapunkat. Ilyenkor az ezen a gépen futó operációs rendszer teljes mértékben megszabhatja, hogy milyen erőforrásokból mennyit használ. Ha egy másik szolgáltatást – például levelezőszervert – szeretnénk emellett futtatni, akkor limitáltabbak a lehetőségeink, hiszen a korábban telepített webszerver már foglal bizonyos erőforrásokat, illetve a program függőségeit és konfigurációs fájljait is telepítettük már, ami esetleg negatívan hat a levelezőszerverünk működésére. Ha mindezt virtualizált környezetben tesszük meg, akkor a topológia megváltozik: a két alkalmazás teljesen elkülönítetten, egymás zavarása nélküli, különböző virtuális gépekben futnak, még magán a fizikai gépen egy úgynevezett hypervisor látja el az erőforrások ütemezésének és kiosztásának (pl. processzoridő, memória) feladatát.

1.2.1. Népszerű virtualizációs technológiák

Mivel a virtualizáció napjainkban nagyon elterjedt technológia, számos olyan megoldás született, mely egyszerűsíti a virtuális gépek üzemeltetését. Ezek közül nagy ismertségnek örvend az Oracle VirtualBox és a VMware Player, azonban ezek a megoldások nem skálázónak annyira jól, mint a továbbiakban tárgyalt társaik, melyek sokkal megfelelőbbek nagyvállalati szerverkörnyezetben való alkalmazásra. Ezek a megoldások lehetőséget biztosítanak a virtuális gépek távoli elérésre, kezelésére, egyszerűbb telepítésükre, valamint szükség esetén elosztott működésükre.

Ilyen nagyvállalati környezetben is kedvelt megoldás például a VMware ESXi, amely egy igen modern hypervisor számos kényelmi funkcióval ellátva (lehetőség van például a rendszer webes felületről való kezelésére és virtuális gépek sablonból való gyors (nagy-ságrendileg 5-10 perc) telepítésére). Egy másik kedvelt megoldás az ESXi-vel ellentétben teljesen ingyenesen, GPLv2-es licenc alatt elérhető XEN hypervisor, mely ugyan kevesebb

kényelmi funkciót tartalmaz, de szintén népszerűségnek örvend széleskörű támogatása, kedvező teljesítménye és szabad szoftver voltából eredő ingyenessége miatt. A XEN a 2014 márciusában kiadott 4.4-es verzió óta stabilan működik együtt a libvirt virtualizációs API-val, amely nagyban megkönnyíti a hypervisorral való kommunikációt a virtuális gépek konfigurálása során. [7] A XEN-hez hasonlóan szabad szoftver licenccel érhető el a Kernel-based Virtual Machine (KVM) is, mely a XEN-nél modernebb megoldásnak tekinthető, és manapság széles körben használják a Linux kernelbe való integráltságának és stabilitásának köszönhetően. Bár maga a KVM nem tartalmaz ilyet, de számos interfész elérhető az ezen keresztül futtatott virtuális gépek kezelésére (például virt-manager), valamint akadnak olyan megoldások is, melyek a KVM-re alapozva nyújtanak szélesebb körű virtualizációs megoldást, ilyen lehet¹ például a Proxmox és a Cockpit.

1.2.2. Virtuális gépek használatának néhány előnye

A virtualizáció számos előnyvel bír a szerverinfrastruktúra karbantartása, könnyű kezelhetősége szempontjából, ebből néhány fontosabbat szeretnék kiemelni.

1.2.2.1. Erőforrások testreszabása

Amikor több tíz vagy több száz szerver üzemeltetéséről van szó, akkor hatványozottan számításba kell vennünk az egyes gépekre jutó költségeket. Virtuális gépek esetén ez azért kedvezőbb egy fizikai gépnél, mert ugyan a nagyvállalati környezetbe szánt szervergépek jelentősen drágábbak a személyes felhasználásra tervezett társaiknál, de akár több tíz virtuális gép egyidejű futtatását is lehetővé teszik. Ezáltal az egy fizikai gépre eső, asztali gépeknél megszokott áramfogyasztáshoz képest jóval nagyobb energiafelvétel sokkal kedvezőbb arányt mutat, ha számításba vesszük a futtatott virtuális kiszolgálók számát is.

Mindezek mellett a nagyvállalati felhasználáshoz tervezett számítógépek jóval hibatűróbbek, hiszen a főbb komponensek redundánsan lettek kialakítva: ezekből az ilyen szererekben legalább kettő van, és a rendszer automatikusan képes detektálni a hardveres hibákat, és ezek figyelembe vételével tovább működni. További előny lehet még hardveres hibák esetén, hogy ezek a számítógépek széles körben támogatják az úgynévezett *hot swappingot*, amely azt jelenti, hogy bizonyos hardverelemek (általában például háttértárrak és memóriamodulok) a számítógép bekapcsolt állapotában, annak működése közben is szolgáltatás nélkül cserélhetőek.

¹A konfigurációtól függően akár többfajta virtualizációs környezet is beállítható, de a KVM az egyik legjobban támogatott.

Előnyös lehet továbbá, hogy a virtuális gépek erőforrásai szabadon módosíthatók, így akár két újraindítás között is változtathatjuk a rendelkezésre álló memória mennyiségét vagy épp a processzormagok számát. Sőt, egyes hypervisorok és operációs rendszerek ezen erőforrások futásidejű megváltoztatását is támogatják bizonyos korlátozások mellett, így gyakorlatilag a fontosabb virtualizált erőforrások is hot swappelhetőnek tekinthetőek.

1.2.2.2. Snapshotok

Egy másik kedvező lehetőség virtuális gépek használata esetén az, hogy úgynevezett snapshotokat készíthetünk róluk. Ezek a snapshotok a gépet egy adott pillanatbeli állapotban reprezentálják, és később ezeket az állapotokat visszaállíthatjuk, ha szükségünk lesz rá. Egyes megoldások a memóriakép mentését is támogatják, így akár egy futó gép is könnyen visszaállítható. A snapshotok készítése hasznos lehet például rendszerfrissítések esetén, így ha valamiféle hiba lép fel a frissítés során, vagy egy adott szoftver nem megfelelően működik azt követően, akkor a frissítés előtt készített snapshotra visszaállva újra teljes értékűen üzemelhet a szerver, amíg a frissítés során fellépő hibát elhárítjuk.

1.2.2.3. Migráció

Részben az előző ponthoz kapcsolódik a virtuális gépek migrációja. Ez a funkció azt jelenti, hogy egy adott fizikai gépről, mely virtuális gépeket futtat (virtual host), készíthetünk egy snapshotot, amit áthelyezhetünk egy másik virtual hostra, és a virtuális gép ezen futtat tovább egyéb újrakonfigurálás nélkül. Lehetőség van azonban a háttértárat tartalmát elhagyva is átmozgatni egy VM-et egy másik hosztra. Ehhez bevett szokás leírófájlok használata, mely egy virtuális gép konfigurációját tartalmazza. A leírófájt egy másik hoszt-gépre áthelyezve ott újra elindíthatjuk a definiált virtuális gépet. Ilyenkor szükség lehet a VM háttértárainak inicializálására, de ettől eltekintve a konfiguráció szabadon hordozható virtual host-ok között. Ilyen migrációra egyes megoldások fejlettebb támogatást is adnak, így akár valós időben, az aktuális terheltség figyelembe vétele mellett automatikusan is áthelyezhetőek virtuális gépek a megadott fizikai hosztok között.

1.2.3. Teljes virtualizáció és paravirtualizáció összehasonlítása

1.2.4. Konténerizáció

1.3. Logikai kötetkezelés

1.3.1. Snapshotok, mentések készítése

2. fejezet

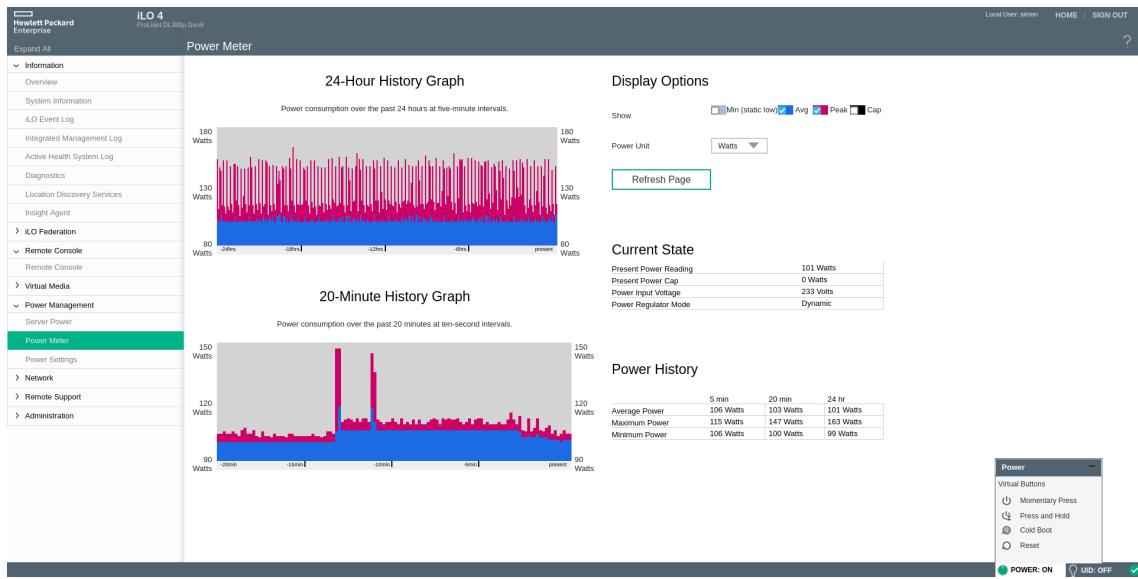
Technológiai áttekintés

2.1. Szervergépek

A szerverek esetében jelentkező, egyénitől nagyban különböző felhasználási körülmények a szerverszámítógépek esetén hardveres szempontból is más felépítést igényelnek. A magas rendelkezésre-állás (high availability, HA) és a modularitás, valamint az ezzel járó könnyű javítások támogatása érdekében az ilyen célra kialakított számítógépek főbb komponensei redundánsak, azaz egy-egy ilyen komponens kiesése nem jelent szolgáltatáskiesést. A meghibásodást a rendszer egyértelműen jelzi magán a gépházon is (általában hibajelző LED-ek segítségével), valamint a menedzsment portjain is. A legtöbb ilyen gép ugyanis rendelkezik egy beágyazott rendszerrel, ami lehetővé teszi a távoli kezelésüket egy webes felületen és SSH-n keresztül még akkor is, ha a szervergép ki van kapcsolva. Ezek lehetőséget biztosítanak a gép legfontosabb mérőszámainak követésére, virtuális kijelző csatlakoztatására, telepítőfájlok felcsatolására, valamint a gép ki- és bekapcsolására.

A fent ismertetett üzemeltetést, karbantartást könnyítő felépítés mellett általában elmondható, hogy az ilyen gépek jelentős része virtualizációra van tervezve – persze ezektől különböző felhasználási módok is jelentkeznek (például fájlszerverek tervezése során a teljesítmény helyett a minél nagyobb tárkapacitásra és adatátvitelre helyezték a hangsúlyt). A dolgozat szempontjából viszont a nagyvállalati környezetben domináló virtualizációs felhasználási terület lesz a lényegesebb, így a továbbiakban az ilyen számítógépekre fogok koncentrálni.

A virtualizációs hosztgépek jellemzője, hogy számos processzorral rendelkeznek, valamint felhasználói szemmel szokatlanul nagy memóriaterülettel bírnak. Ki fog derülni azonban, hogy 12-24 processzormag és akár több száz gigabyte RAM is szűkös erőforrássá válhat egy virtuális gépeket futtató számítógép esetében, hiszen gyakorlatilag itt egyetlen



2.1. ábra. Szervergép fogyasztásának grafikonja egy HPE számítógép távoli menedzsment felületén. Vegyük észre a jobb alsó sarokban megjelenő menüt, amivel lehetőségünk van a gép kikapcsolására és újraindítására is.

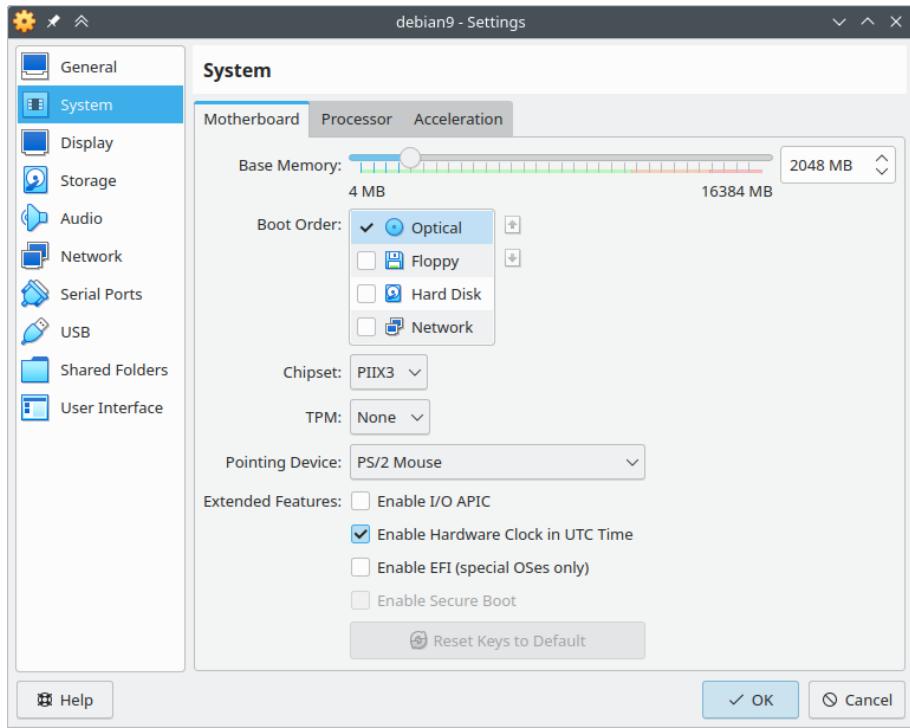
szervernek kell elbírnia akár több tíz számítógép terhelésével is. Ezek mellett általában több (8-24) háttértár-foglalattal is rendelkeznek, melyekhez hardveres RAID-támogatást is adnak.

2.2. Virtualizáció

2.2.1. Áttekintés

A virtualizáció egy olyan technológia, amely lehetővé teszi, hogy egy fizikai számítógépen (az úgynevezett virtuális host-on) több, akár a hosztgép rendszerétől eltérő operációs rendszert futtassunk. Ehhez szükség van egy hypervisor-ra, ami az operációs rendszer legfőbb virtualizációt támogató komponense [5]. Ez a szoftver közvetlenül a szerver hardverén fut, és ez biztosítja a virtuális gépek számára szükséges hardveres erőforrásokat virtualizált hardverinterfészeken keresztül. Szintén a hypervisor felelős az erőforrások kezeléséért, tehát ennek a komponensnek kell biztosítania a virtuális gép számára beállított mennyiségű memória, processzor és háttértár rendelkezésre állását is. Ezek a beállítások a legtöbb modern hypervisor esetében elvégezhetők grafikus felületen is, erre ad példát a 2.2 ábra.

Napjainkban a legelterjedtebb hypervisorok közé tartozik a vmWare ESXi, a XEN és a KVM, melyek részletesebb bemutatását a 2.2.3 alfejezet tartalmazza.



2.2. ábra. Virtuális gép beállításainak részlete a VirtualBox virtuálizációs szoftverben.

A virtualizáció számos előnyvel járhat az infrastruktúra és a kiszolgálni kívánt alkalmazások szempontjából. Az egyik legnagyobb ilyen előny például, hogy a virtuális gépek egymástól izoláltan futnak, azaz nincs közvetlen kapcsolat közöttük, ami biztonsági és kezelési, tesztelési szempontból (pl. egy adott csomag vagy szoftver kipróbálásához készíthetünk egy teszt virtuális gépet, amit egyszerűen törölhetünk a teszt végeztével; a telepített program eltávolítása hagyományos környezetben futtatva sokkal körülményesebb lenne) is kedvező lehet. Hasonlóan előnyökkel jár, hogy a legtöbb modern hypervisor lehetőséget biztosít bizonyos erőforrások úgynevezett hot swap-pelésére. Ez azt jelenti, hogy egyes komponenseket (pl. memória) úgy is bővíthetünk, hogy a rendszert nem szükséges leállítanunk.

2.2.2. Paravirtualizáció és teljes virtualizáció

2.2.3. Virtualizációs lehetőségek összehasonlítása

[TODO: vmware a könnyű kezelhetőség és jó támogatás miatt magasan az élen áll; XEN, KVM]

2.3. Logikai kötetkezelés

Mind a fizikai, mind a virtuális gépek esetén szükség lehet háttértáraakra az adatok persistens tárolása érdekében. Hagyományos particionálási megoldásokkal hamar nehezen kezelhetővé válhatnak a különböző csatolási pontok és a virtuális gépek számára kiosztott kötetek. Az ilyen problémák elkerülésére jött létre a logikai kötetkezelés, mely a tárhely-virtualizáció egy formája. A logikai kötetkezelésnek több implementációja létezik. Ezek közül jelenleg a Linux kernelben elérhető Logical Volume Manager (LVM)-et fogom részletesen ismertetni.

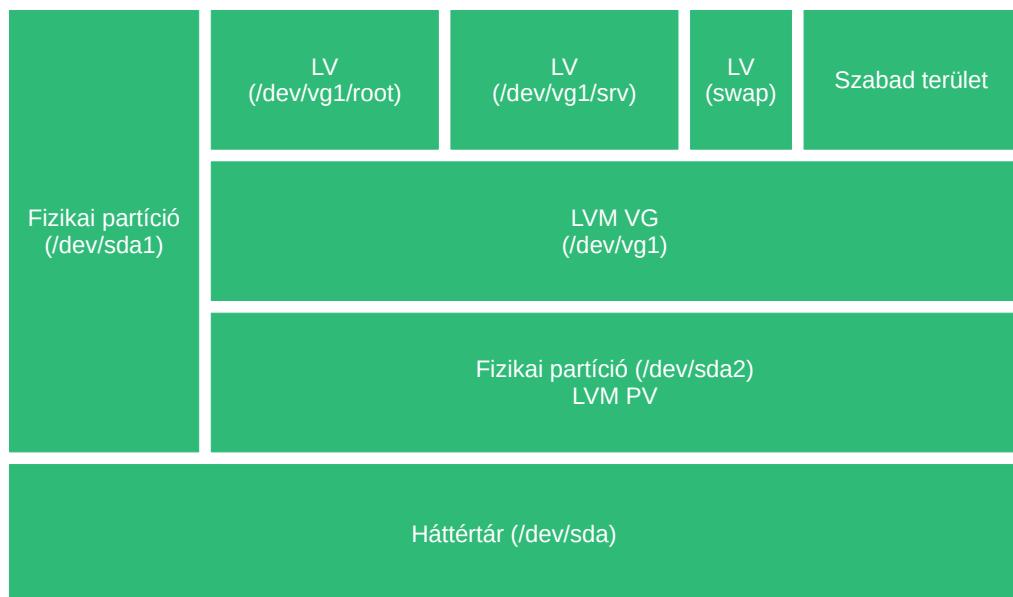
A Linux logikai kötetkezelője három lényegi rétegből áll: a fizikai kötetből (physical volume, PV), a kötetcsoportból (volume group, VG) és a logikai kötetekből (logical volume, LV). Ezt a felépítést a 2.3 ábra szemlélteti egy egyszerű LVM-konfiguráció keretében. Lehetőség van ennél összetettebb kötetkiosztás létrehozására is, például egy kötetcsoport több fizikai kötetből is állhat, amik akár külön háttértáron is lehetnek, sőt, RAID-csoportot is megadhatunk egy LVM-partíció alapjául. Ezen megoldások használata azonban sok hátránya járhat (pl. diszkhiba esetén nehezebb visszaállítani a partíciót), ezért ennek használata alapvetően nem ajánlott [2].

Az LVM tehát úgy épül fel, hogy egy vagy több háttértáron létrehozunk hagyományos fizikai partíciókat, melyek az LVM PV-k alapjául fognak szolgálni. Ezt követően létrehozzuk a kötetcsoportokat az általuk használandó LVM fizikai kötetek megadásával. Az így létrejött csoportban már tudunk létrehozni logikai köteteket, amíg van szabad hely a VG-ben.

Láthatjuk, hogy az LVM-kötetek használata kezdetben több feladattal jár, mint a hagyományos partíciók esetében, azonban hosszabb távon számos előnnyel jár. Talán a logikai kötetkezelés legnagyobb előnye, hogy szabadon allokálhatunk tárterületet a létrehozott köteteknek: ha azt tapasztaljuk, hogy az egyik köteten kevés a szabad hely, akkor fájlrendszerrel függően elég lehet akár egy parancs kiadása is ennek kiterjesztéséhez. Lényeges, hogy a hagyományos partíciók használatával ellentétben a logikai kötetkezelés használatakor figyelmen kívül hagyhatjuk a partíciók elhelyezkedésének sorrendjét, így nem szükséges figyelembe vennünk, hogy az adott partíció előtt vagy után van-e szabad tárterület. A megnövelt kötet helyes fizikai háttértárra képzéséről a logikai kötetkezelő fog gondoskodni számunkra. Fontos megjegyezni, hogy a kötetbővítés online is elvégezhető, azaz nem szükséges a kötetet lecsatolni a gépről az átméretezéshez. Ez különösen fontos lehet például a root (/) partíció növelése során, hiszen ezt csak a számítógép leállítása mel-

lett tudjuk biztonságosan lecsatolni. Előállhat olyan helyzet is, hogy egy másik (nem root) partíciót kell online átméretezniünk, például ha azt tapasztaljuk, hogy egy adatbázisszerveren hirtelen nagy mértékben nőtt a tárolt adat mérete. Ilyenkor nincs lehetőség a szerver leállítására, hiszen ez esetben az alkalmazások nem tudnák használni az adatbázist a leállás idejére. Az ehhez hasonló helyzetekre is jó megoldást nyújt a logikai kötetkezelő egy megfelelő, online átméretezést támogató fájlrendszer (pl. XFS, Btrfs) használata mellett. Érdemes megjegyezni, hogy bár az LVM és például a Btrfs-fájlrendszer nyújt támogatást a növelésen kívül a fájlrendszer méretének csökkentésére is, ez a művelet általában nem biztonságos, és adatvesztéshez vezethet. Emiatt érdemes eleinte csak kisebb tárterületet adni a köteteinknek, hiszen kiterjeszteni sokkal egyszerűbb őket, mint csökkenteni a méretüket. Ennek megkönnyítésére is ad lehetőséget az LVM, megadhatjuk, hogy egy kötet egy bizonyos arányú tárhelyhasználat után automatikusan bővüljön, így elkerülve annak betelését.

Az LVM hasznos funkciói közé tartozik még a kötetpillanatképek (volume snapshots) készítésének lehetősége. Ez azt jelenti, hogy a kötetkezelő képes az adott kötet adott pillanatbeli helyzetének rögzítésére, és erre a verzióra szükség szerint visszaállhatunk (rollback). Ez hasznos lehet például nagyobb konfigurációs változások eszközölése esetén, gyorsan változó adatokkal dolgozó rendszerek (pl. adatbázisszerver) biztonsági mentéseinek készítése során, illetve rendszerfrissítések előtt.¹



2.3. ábra. Egyszerű LVM-kötetkezelési hierarchia.

¹Egyes eszközök és operációs rendszerek (pl. openSUSE-verziók a snapper-rel (<https://doc.opensuse.org/documentation/leap/reference/html/book-reference/cha-snapper.html>) automatikusan készítnek snapshotot a frissítések telepítése előtt, így hiba esetén visszaállhatunk a frissítés előtti verzióra.

2.4. OS-lehetőségek

Egy nagyvállalati informatikai infrastruktúrában nagy szerepe van a választott operációs rendszernek is, ugyanis nem mindegy, hogy a több száz számítógépből álló rendszerünket mennyire hatékonyan tudjuk karban tartani, egy kritikus biztonsági frissítést milyen hamar tudunk telepíteni az érintett eszközökre, és probléma vagy különleges igény esetén milyen támogatásra számíthatunk a szoftvereinket illetően. Ezeket a szempontokat figyelembe véve manapság elsősorban a Debian, Ubuntu, Red Hat Enterprise Linux és SUSE Linux Enterprise disztribúciók közül választanak a vállalatok.

A Debian stabilitása miatt népszerű választás elsősorban kisebb (néhány tíz gépből álló) infrastruktúrák esetében, viszont a stabilitás az elérhető csomagok verziójának rovására megy, általában a legújabbnál néhány verzióval régebbi csomagokat szállítanak a disztribúcióval. A Debian előnye, hogy teljesen ingyenesen elérhető, és bár nincs hozzá hivatalos támogatás, harmadik félről vásárolhatunk ilyen szolgáltatást.

Az Ubuntu egy Debian-alapú operációs rendszer, melyet a Canonical Ltd. fejleszt, és vállalati támogatást is nyújt az OS-hez amellett, hogy az alapverzió ingyenesen érhető el. Előnye, hogy mivel mind szerver, mind pedig asztali környezetben elterjedt rendszer, számtalan projekt és gyártó adja ki a szoftvereit Ubuntu rendszerekre.

A Red Hat és a SUSE Linux-verziók már inkább egy magasabb kategóriát céloznak meg: fő célközönségük a több száz, illetve több ezer gépes környezetet üzemeltető vállalatok, és a fent említett két disztribúciónál alapesetben (a legkisebb támogatási csomagban) is szélesebb körű támogatást biztosítanak az operációs rendszerekhez. Kiemelendő, hogy ez a két disztribúció egyedülálló a biztonság területén: számos biztonsági tesztnek vetették alá őket különböző szervezetek (köztük például kormányzatok és IT-biztonságra specializálódott cégek is), melyeket követően a kereskedelmi forgalomban lévő Linux-disztribúciók közül a legmagasabb minősítéseket és tanúsítványokat kapták meg ezek a rendszerek [1] [4].

[TODO] Debian/Ubuntu, RedHat, SuSE; csomagkezelők, biztonság

2.5. Eszközmenedzsment

2.5.1. Ansible és Salt

2.6. Monitoring

3. fejezet

Saját munka bemutatása

3.1. Kialakítani kívánt környezet meghatározása

Dolgozatomban egy kisebb léptékű, de a fontosabb elvek ismertetését kellő mértékben lehetővé tevő tesztkörnyezetet fogok kialakítani és részletesen bemutatni. A tesztkörnyezetben egy fizikai gépen (virtual host, dom0) fogok virtuális gépeket kialakítani a KVM hypervisor segítségével. Ezen környezet célja, hogy betekintést engedjen a nagyvállalati környezetek kialakításának fontosabb lépéseihe. Ahogy arról a bevezetés során szó volt, ilyen környezetekben nem ritka, hogy több száz vagy több ezer gépet kell kezelnünk. Nyilvánvaló, hogy ilyen nagyságrendű rendszert nem lehet kizárolag manuális megoldásokkal kezelni, nem várhatjuk el a rendszergazdáktól és az üzemeltetést végző mérnököktől, hogy egy-egy frissítést minden gépen külön-külön telepítsenek: ehhez sok esetben nem is adottak a lehetőségek, hiszen a legtöbb ilyen számítógép egy adatközpontban fut, ahova a biztonsági előírások miatt nem tudna minden munkatárs bejutni, továbbá körülményes lenne az egyes számítógépekhez külön-külön perifériákat csatlakoztatni, amire például a nagyobb operációsrendszer-verziótáltások során szükség lehetne az úgynevezett *offline upgrade*-ek miatt – ilyen helyzetekben a távoli elérés (pl. SSH, VNC) sem jelent megoldást.

Mindezen okok miatt a dolgozatomban egy infrastruktúramenedzsment eszközt fogok használni, amivel növelhető az egyes kiszolgálók karbantartásának hatékonysága mind fizikai, mind pedig virtuális gépek esetén.

3.2. Fizikai gép ismertetése

3.3. Operációs rendszer

Értekezésemben nagy szerepe lesz a választott operációs rendszereknek, hiszen ezek fognak a virtualizációs rendszer alapjául szolgálni, valamint képesnek kell lennünk a gépek távoli menedzsmentjére is, így mindenkorban olyan megoldásra van szükség, amely jól támogatott a választott infrastruktúramenedzsment eszköz által. Fontos szempont volt továbbá, hogy a tesztkörnyezet a lehetőségekhez mérten jól reprezentálja a nagyvállalati környezetben használatos rendszereket, így sok olyan OS-verzió kikerült a lehetőségek közül, amelyek ugyan népszerűek például asztali megoldásként, de egyes nagyvállalati szoftverek (legyen az adatbázismotor, vagy bizonyos eszközvezérlők, driverek) hivatalosan nem támogatottak rajtuk. Emiatt az operációs rendszerek kiválasztása során körültekintően jártam el, több Linux-disztribúció is szóba került, az ezekről született konklúziót itt foglalom össze néhány mondatban.

3.3.1. Debian, Ubuntu

3.3.2. SUSE

3.4. Gépmenedzsment: Salt

3.4.1. Más hasonló megoldások

3.5. Monitoring

3.6. Továbbfejlesztési lehetőségek

Köszönetnyilvánítás

Ez nem kötelező, akár törölhető is. Ha a szerző szükségét érzi, itt lehet köszönetet nyilvánítani azoknak, akik hozzájárultak munkájukkal ahhoz, hogy a hallgató a szakdolgozatban vagy diplomamunkában leírt feladatokat sikeresen elvégezze. A konzulensnek való köszönetnyilvánítás sem kötelező, a konzulensnek hivatalosan is dolga, hogy a hallgatót konzultálja.

Szójegyzék

hot swap Számítógép-komponensek eltávolítása vagy hozzáadása egy futó rendszerhez (annak leállítása vagy újraindítása nélkül). 7

hypervisor Az a szoftver, amely koordinálja a virtuális gépek és az azokat futtató fizikai gép hardvere közötti interakciót. 7

Acronyms

HA high availability. 6

LVM Logical Volume Manager. 7–9

PV physical volume. 8

VG volume group. 8

Irodalomjegyzék

- [1] Red Hat: Red Hat Adds Common Criteria Certification for Red Hat Enterprise Linux 8 (2024. április 24.). <https://www.redhat.com/en/about/press-releases/red-hat-adds-common-criteria-certification-red-hat-enterprise-linux-8>.
- [2] Red Hat: Red Hat Enterprise Linux: Logical Volume Manager Administration (2024. április 22.). https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/logical_volume_manager_administration/lvm_components.
- [3] Seagate: What is a 3-2-1 backup strategy? (2024. április 7.). <https://www.seagate.com/gb/en/blog/what-is-a-3-2-1-backup-strategy/>.
- [4] SUSE: SUSE Certifications and Features (2024. április 24.). <https://www.suse.com/support/security/certifications/>.
- [5] SUSE: SUSE Linux Enterprise Server 15 SP5 – Virtualization Guide (2024. április 11.).
https://documentation.suse.com/sles/15-SP5/pdf/book-virtualization_en.pdf.
- [6] Wikipedia: Standard raid levels (2024. április 7.). https://en.wikipedia.org/wiki/Standard_RAID_levels.
- [7] XEN Project: Xen project 4.4 release notes (2024. március 20.). https://wiki.xenproject.org/wiki/Xen_Project_4.4_Release_Notes.
- [8] Yev Pusin, Backblaze: The 3-2-1 backup strategy (2024. április 7.). <https://www.backblaze.com/blog/the-3-2-1-backup-strategy/>.