

Önálló laboratórium beszámoló

Távközlési és Médiainformatikai Tanszék

készítette: Réthelyi Bálint
rethelyibalint@gmail.com

neptun-kód: IZUM0B

ágazat: Mérnökinformatikus szak

konzulens: Dr. Simon Csaba
simon@tmit.bme.hu

konzulens: Dr. Maliosz Markosz
maliosz@tmit.bme.hu

Téma címe: Proxmox alapú virtualizációs klaszter kiépítése a HSNLabnál

Feladat:

Proxmox alapú virtualizációs klaszter kiépítése és üzemeltetésének megkönnyítése a HSNLabnál.

Tanév: 2021/22. tanév, II. félév

Bevezetés

Az önálló laboratóriumomat a TMIT-en a HSNLab-nál végeztem. A labornak a különböző kutatásokhoz van négy szervere. Ezeket a szervereket használják Kubernetes, valamint docker konténerek futtatására, és ezeken a szervereken keresztül érik el a tanszéki belső hálózatot is.

Amikor átvettem a szerverek üzemeltetését, a négy gép úgynevezett bare-metal szerver volt, azaz a fizikai szerveren futott egy operációs rendszer és a felhasználók azt használták. A hosztokon Ubuntu 18.04 futott.

Problémák a jelenlegi megoldással

Míg első elgondolásra a bare metal megoldás kényelmesnek hangzik üzemeltetési szempontból, erről kiderült, hogy azért bőven akad vele probléma.

A laborban sokan dolgoznak, csak a BSc és MSc hallgatók száma meghaladja a százat. A laborban az volt a szokás, hogy mindenki a root felhasználóhoz kapott hozzáférést, ezzel a docker használatához szükséges jogosultságok kiosztására nem volt szükség.

Ez a megoldás sajnos nehezebb auditálhatóságot okozott, hiszen minden felhasználó a root felhasználó nevében végezte a munkáját, így mindent a root felhasználó csinált.

Minden felhasználónak különböző csomagverziók és azok függőségeire volt szüksége, ezért ezek karbantartása és kezelése is problémákat okozott, hiszen mindenkinek a fizikai gépen kellett dolgoznia.

Összességében arra jutottam, hogy a virtualizáció lenne a megoldás ezekre a problémákra, melyet egy hypervisorral (vagy hiperfelügyelővel) terveztem megoldani.

Miért jó a virtualizáció a bare-metal szerver helyett?

Mi is pontosan egy bare-metal szerver?

A bare-metal szerver a fizikai szervert jelenti, melyet általában egy kliens/felhasználó használ. Sokkal nagyobb számítási kapacitással rendelkezik, mint egy-egy virtuális gép, hiszen hozzáférünk a fizikai hoszt összes erőforrásához, beleértve a memóriát, a processzort, a diszkeket és a sávszélességet is. Előny még emellett, hogy teljes felügyeletünk lehet a fizikai gép felett, mind hardveresen, mind szoftveresen.

Mi az a virtualizáció?

A virtualizáció egy hosztíng megoldás, ami több virtuális gép párhuzamos futtatását teszi lehetővé egy fizikai eszközön. Lehetőséget biztosít a fizikai gép erőforrásainak *feldarabolására*, és ezek kiosztására a virtuális gépek között, így azok ugyanazon erőforrás részein, *darabkái*n tudnak megosztani.

Virtuális gépek használata elősegíti a felhasználók könnyebb auditálhatóságát, valamint minden felhasználónak egyedi, elszeparált környezetet biztosít, amit a saját igényeinek megfelelően alakíthat.

Ha a virtualizációs megoldást választják, akkor a virtuális gépek kiosztását és menedzselését rábízják egy úgynevezett hypervisorra (magyarul hiperfelügyelőre).

Mi az a hypervisor?

A hypervisor a virtualizációt lehetővé tevő szoftverek kulcsfontosságú része. A hypervisor feladata a virtuális gépek (vendéggépek) előkészítése és a virtuális gépek által elért hardver funkciók elkapása, feldolgozása.

A hypervisorok egy virtualizációs réteget hoznak létre az által, hogy felügyelik, hogy a virtuális gépeken futó folyamatok mely fizikai erőforrásokhoz férnek hozzá.

A gépet, amelyre a hypervisort telepítjük, gazdagépnek nevezzük, szemben a rajtuk futó virtuális vendéggépekkel.

A hypervisorok a fizikai beosztásán kívül biztosíthatnak emulált hardvereket (merevlemez, egér, képernyő, hálókártya, stb.), amelyek használatával a vendéggépek úgy viselkedhetnek, mintha fizikai hardveren futnának.

A Virtual Machine (virtuális gép) – a továbbiakban VM – szempontjából nincs különbség a fizikai és a virtualizált környezet között. A vendéggépek nem tudják, hogy a hypervisor virtuális környezetben hozta létre őket. Sem azt, hogy megosztják a rendelkezésre álló számítási teljesítményt. A VM-ek változatlanul az őket működtető hardveren futnak, így teljes mértékben függnék annak stabil működésétől.

A hypervisoroknak tradicionálisan két típusa van. Az első típust bare-metal hypervisoroknak is szokták hívni, míg a másodikat hosztolt hypervisoroknak. (Simic)

Első típusú hypervisor

A bare-metal hypervisor egy olyan szoftverréteg, amelyet közvetlenül a fizikai kiszolgálóra telepítünk.

Nincs közte szoftver vagy bármilyen operációs rendszer, innen a bare-metal hypervisor (magyarul csupasz metál) elnevezés. Az 1-es típusú hypervisor kiváló teljesítményt és stabilitást biztosít, mivel nem egy operációs rendszeren belül fut.

Az 1-es típusú hypervisor egy nagyon alapszintű operációs rendszer, amelyen virtuális gépeket lehet futtatni. A fizikai gép, amelyen a hypervisor fut, csak virtualizációs célokat szolgál. Másra nem használható.

Második típusú hypervisor

Az ilyen típusú hypervisor egy fizikai gép operációs rendszerén belül fut.

Ezért nevezzük a 2. típusú hypervisorokat hosztolt hypervisoroknak. Az 1. típusú hypervisorokkal szemben, amelyek közvetlenül a hardveren futnak, a hosztolt hypervisorok alatt egy szoftverréteg található. Ebben az esetben a következőkkel rendelkezünk:

- Egy fizikai gép.
- A hardverre telepített operációs rendszer (Windows, Linux, macOS).
- Az operációs rendszeren belül egy 2. típusú hypervisor szoftver.
- A vendég virtuális gépek tényleges példányai.

Ebben az esetben a fizikai hoszton is tudjuk kezelni, menedzselni a virtuális gépeinket.

A modern hypervisor

Manapság rájöttek, hogy a bare-metal hypervisor jó tulajdonságai mellett kényelmes, ha az operációs rendszer teljes értékű. Ezért a gyártók beleépítették az operációs rendszerekbe a hypervisor modult, ezzel elérve, hogy legyen egy tradicionális operációs rendszerünk, annak minden pozitív tulajdonságával és kényelmi funkciójával, és a hypervisor is kernel szintű legyen, ne legyen még egy plusz réteg az operációs rendszer fölött.

A teljesség igénye nélkül ilyen megoldások az alábbiak:

- VMware ESXi
- Hyper-V
- Linux + KVM
- bhyve (FreeBSD)
- Hypervisor framework (Apple macOS)
- Proxmox

Modern hypervisorok kezelése

Ha elindítunk egy fizikai kiszolgálót, amelyen egy hypervisor van telepítve, egy parancsszerű képernyő jelenik meg. Ha egy monitort csatlakoztatunk a szerverhez, akkor a hardver és a hálózat néhány részletét láthatjuk. Ez

általában a CPU típusából, a memória mennyiségéből, az IP-címből és a MAC-címből áll.

Általában ezek csak egyszerű szervert konfigurációt tesznek lehetővé.

Ez a dátum és az idő, az IP-cím, a jelszó stb. megváltoztatásából áll. A virtuális példányok létrehozásához egy másik gépen beállított kezelőkonzolra van szükség. A konzol segítségével csatlakozhatunk a szerveren lévő hypervisorhoz, és kezelhetjük a virtuális környezetet.

A kezelési konzol lehet webalapú vagy különálló szoftvercsomag, amelyet telepíthetünk arra a gépre, amelynek távoli kezelését szeretnénk.

Az egyik elvégezhető művelet a virtuális gépek fizikai kiszolgálók közötti manuális vagy automatikus áthelyezése. Ez a mozgás a VM adott pillanatban fennálló erőforrásigényén alapul, és a végfelhasználókra gyakorolt hatás nélkül történik. Ugyanez a folyamat történik akkor is, ha egy hardverdarab vagy egy egész szerver meghibásodik. A megfelelően konfigurált kezelőszoftver a virtuális gépeket egy működő szerverre helyezi át, amint probléma merül fel. Az észlelési és helyreállítási eljárás automatikusan és zökkenőmentesen zajlik.

A hypervisorok egyik legjobb tulajdonsága, hogy lehetővé teszik a fizikai erőforrások túlkiosztását.

A hypervisorokkal több erőforrást rendelhetünk a virtuális gépekhez, mint amennyi rendelkezésre áll. Ha például a kiszolgálón 128 GB RAM van, és nyolc virtuális gépet használunk, mindegyikhez 24 GB RAM-ot rendelhetünk. Ez összesen 192 GB RAM-ot jelent, de maguk a VM-ek valójában nem fogják elfogyasztani a fizikai kiszolgáló mind a 24 GB-ját. A VM-ek azt hiszik, hogy 24 GB-ot használhatnak, holott valójában csak annyi RAM-ot használnak, amennyi az egyes feladatok elvégzéséhez szükséges.

A hypervisor csak annyi erőforrást rendel ki, amennyi szükséges ahhoz, hogy egy példány teljes mértékben működőképes legyen.

Modern hypervisorok áttekintése

VMware ESXi

Az ESXi egy BSD alapú operációs rendszer, melynek magja a VMkernel, mely a FreeBSD kernelén alapszik és ebbe építették bele a hypervisort.

Nagyvállalati környezetben a legtöbbször ezt a megoldást a vSphere-el együtt használják, mely lehetővé teszi több ESXi hoszt klaszterizálását.

Korábbi munkáim során találkoztam már a megoldással, és a tanszéki környezethez feleslegesen bonyolultnak és túlságosan robusztusnak ítélt meg, ezért elvettem a használatát.

Hyper-V

A Hyper-V a Microsoft hypervisor megoldása, mely mind a fogyasztói Windows, mind Windows Server operációs rendszereken elérhető. Természetesen az átlag felhasználók számára jóval kevesebb funkció érhető el, mint a szerveres környezetben.

A Hyper-V a Microsoft által készített NT kernel része, így Windowson lehetővé teszi virtuális gépek futtatását.

Ezzel a megoldással is találkoztam már korábban, és mivel a tanszéki környezetben inkább Linuxos környezetre van szükség, feleslegesen bonyolultnak éreztem egy Windowsos réteg bevonását az ökoszisztémába.

Linux plusz vanilla KVM

A KVM, azaz Kernel-based Virtual Machine (kernel alapú virtuális gép) egy virtualizációs infrastruktúra a Linux rendszermagba integrálva. A modul betöltésekor a Linux kernel egy vékony része Hypervisorra válik. A KVM natív (hardver-segített) virtualizációt támogat.

A KVM egy nyílt forráskódú megoldás, mely egy betölthető kernelmodulból, a `kvm.ko`-ból, amely az alapvető virtualizációs infrastruktúrát biztosítja, és egy processzorspecifikus modulból, például a `kvm-intel.ko` vagy `kvm-amd.ko` modulból áll.

Ez egy egyszerű API-t biztosít, mely segítségével könnyű virtuális gépeket futtató programokat írni, például ilyen a QEMU is.

Ezen a megoldáson (QEMU+KVM) alapszik a Proxmox VE is, amelyet végül a tanszéki rendszerek hypervisorának választottam.

Proxmox VE

A Proxmox VE egy teljes körű, nyílt forráskódú, Debian alapú szervermenedzsment platform virtualizációhoz. Szorosan integrálja a KVM hypervisort és a Linux Containers (LXC), a szoftveresen definiált tárolási és hálózati funkciókat egyetlen platformon. Az integrált webalapú felhasználói felülettel könnyedén kezelhetők a VM-ek és konténerek, a klaszterek magas rendelkezésre állása vagy az integrált katasztrófa-helyreállítási eszközök (disaster recovery tools).

Proxmox VE telepítése

Beállítások kiválasztása

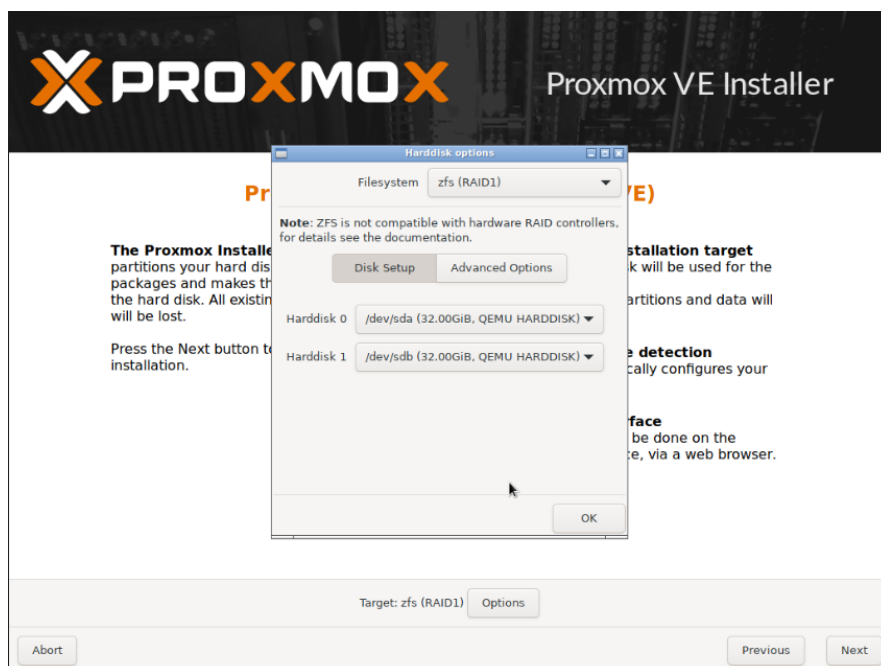
Fájlrendszer beállítása

Szerettem volna, ha szoftveres raid-et használna a szerver `zfs`-el, mivel a `zfs` fájl szinten kezeli az adatokat, így sokkal több lehetőség áll rendelkezésre az adatok védelme, sebességnövelés, stb. érdekében. Hardveres raid-el szemben a szoftveres raid nincsen hardverhez kötve, így meghibásodás esetén a merevlemezeket át lehet tenni egy másik gépbe, ahol szoftveresen ugyanúgy össze lehet rakni a raid-et, és le lehet menteni az adatokat, valamint így a vendor lock-in (gyártókhöz kötés) se áll fent. A szoftveres raid rendszeresen kap frissítéseket, így az idővel jobb, hatékonyabb lehet, valamint biztonsági hibákat is lehet rajtuk javítani, ellentétben egy hardveres raid vezérlővel. (Dr. Korn)

Ezen kívül a `zfs` támogat:

- `cow` (copy on write)-ot (magyarul: írás közbeni másolás)
- transzparens tömörítést
- `thin allocated` (magyarul: vékonyan allokált) virtuális diskek létrehozását
- snapshotokat (pillanatképeket) és azok hálózaton való átküldését

Az 1. ábrán láthatóak a fájlrendszer beállításai.

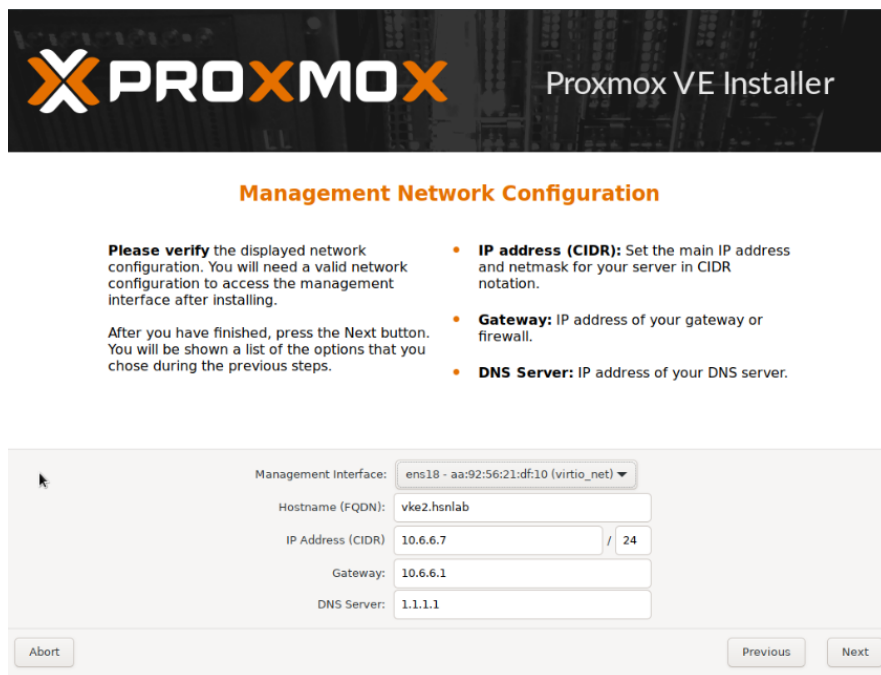


1. ábra: Fájrendszer beállítása

Hálózat beállítása

Először szeretném, hogy a hosztot el lehessen érni rögtön a tanszéki hálózatról, valamint legyen hozzáférése az internethez is, így a korábbi interfészét szeretném megtartani, valamint a korábbi interfészbeállításokat is. Azaz beállítom menedzsment interfésznek a korábbi interfészt (ez jelen esetben az ens18 interfész). Hosztnévnek a vke2.hsnlab-ot veszem fel (ez beírásra kerül a /etc/hosts fájlba, valamint ez alapján lesz beállítva a hoszt hosztneve). IP címnek beállítom a korábban használt címet: 10.6.6.7/24 és átjárónak szintén a korábbi átjárót: 10.6.6.1.

A 2. ábrán a hálózati beállítások láthatóak.



The image shows the Proxmox VE Installer's Management Network Configuration screen. At the top, the Proxmox logo and 'Proxmox VE Installer' text are displayed. Below this, the title 'Management Network Configuration' is centered. The screen contains two columns of text. The left column provides instructions: 'Please verify the displayed network configuration. You will need a valid network configuration to access the management interface after installing.' and 'After you have finished, press the Next button. You will be shown a list of the options that you chose during the previous steps.' The right column lists three configuration items: 'IP address (CIDR): Set the main IP address and netmask for your server in CIDR notation.', 'Gateway: IP address of your gateway or firewall.', and 'DNS Server: IP address of your DNS server.' Below the text is a form with the following fields: 'Management Interface' (a dropdown menu showing 'ens18 - aa:92:56:21:df:10 (virtio_net)'), 'Hostname (FQDN):' (a text box with 'vke2.hsnlab'), 'IP Address (CIDR):' (a text box with '10.6.6.7' and a netmask box with '24'), 'Gateway:' (a text box with '10.6.6.1'), and 'DNS Server:' (a text box with '1.1.1.1'). At the bottom, there are three buttons: 'Abort', 'Previous', and 'Next'.

2. ábra: Hálózat beállítása

A hardveres RAID vezérlő átállítása HBA módba

Mivel korábban úgy döntöttem, hogy a Proxmoxon szoftveres raid-et fogok használni, ezért át kellett kapcsoljam a hardveres RAID vezérlőt HBA módba.

SAS vezérlők módjai

A szerverekben több merevlemez vagy SSD együttes használatához úgynevezett SAS vezérlőt alkalmaznak (Serial Attached SCSI), mely segítségével sorosan lehet a merevlemezeket csatlakoztatni.

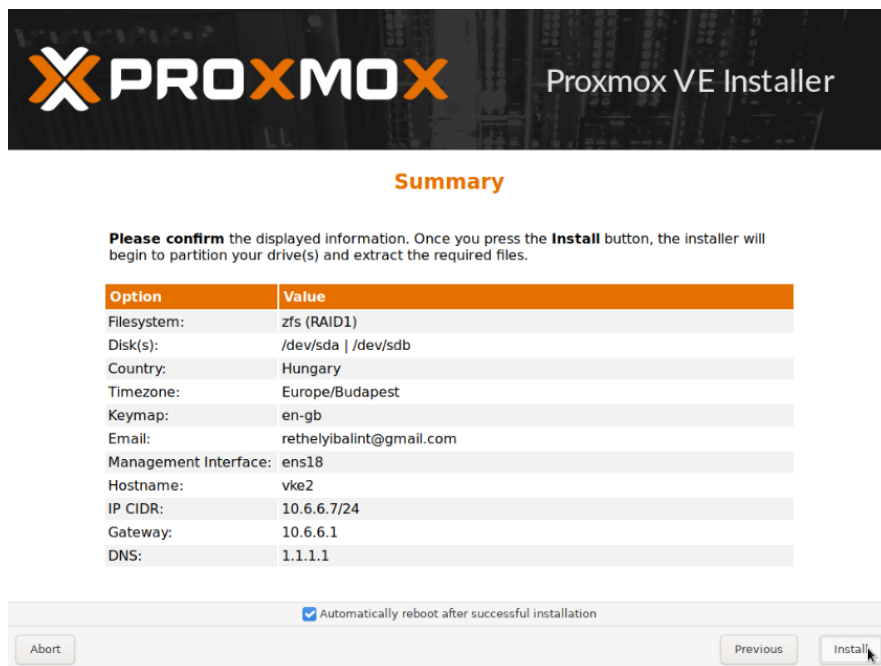
Ezeknek a SAS vezérlőknek általában két módja van, a RAID mód és a HBA mód. (*What are IT mode, HBA mode, RAID mode in (SAS) Controllers?*)

RAID módban, azaz Redundant Array of Independent Disks (magyarul: független lemezek redundáns tömbje), az operációs rendszer nem fogja látni a független lemezeket, hanem egy virtuális tömbként, eszközként fogja azt kezelni.

HBA módban (Host Bus Adapter) a vezérlő átadja a lemezeket az operációs rendszernek, azaz az operációs rendszer látni fogja az összes független lemezt.

Telepítés

A Proxmox telepítése innentől kezdve egy *Next, Next, Finish* jellegű feladat, a telepítés előtt még leellenőriztem a beállításokat, melyek a 3. ábrán láthatóak.



Summary

Please confirm the displayed information. Once you press the **Install** button, the installer will begin to partition your drive(s) and extract the required files.

Option	Value
Filesystem:	zfs (RAID1)
Disk(s):	/dev/sda /dev/sdb
Country:	Hungary
Timezone:	Europe/Budapest
Keymap:	en-gb
Email:	rethelyibalint@gmail.com
Management Interface:	ens18
Hostname:	vke2
IP CIDR:	10.6.6.7/24
Gateway:	10.6.6.1
DNS:	1.1.1.1

☒ Automatically reboot after successful installation

Abort Previous **Install**

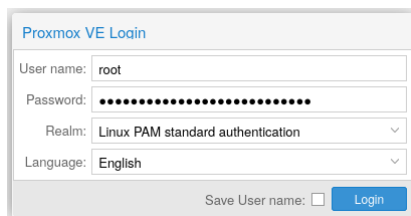
3. ábra: Beállítások ellenőrzése

A Proxmox üzemeltetése

A Proxmoxot általában két módon lehet elérni, konzolosan, vagy pedig a webes interfészen keresztül.

Webes interfész

A webes interfészt (Proxmox GUI) a fizikai gép IP címén lehet elérni, a 8006-os porton. Fontos a biztonságos kapcsolat (https), mert anélkül nem tölt be. Így a korábban telepített szervert jelenleg a `https://10.6.6.7:8006` címen lehet elérni (tanszéki hálózatról). Belépéskor a Linux PAM autentikációt kell választani jelen esetben, ebben az esetben a hoszt gépen létrehozott root felhasználóval tudunk belépni, ezt a 4. ábrán láthatjuk.



Proxmox VE Login

User name: root

Password:

Realm: Linux PAM standard authentication

Language: English

Save User name: ☐ Login

4. ábra: Proxmox GUI belépés

Konzolos (CLI) interfész

A konzolos interfészt vagy fizikai hozzáféréssel érjük el, a hosztra csatlakoztatott billentyűzet és monitor kombinációval, vagy a gyakoribb megoldás, hogy ssh-n keresztül csatlakozunk a hosztra.

Mivel korábban pont azért választottam a Proxmoxot, hogy kapjak egy tradicionális linux shellt (héjat), ezért a megszokott linuxos parancsokon kívül van pár extra, Proxmox specifikus parancs is.

VM készítése (automatizálás)

A hypervisoroknak a legfontosabb funkcionalitása természetesen a virtuális gépek készítése, menedzselése. Erre a Proxmox nyújt webes felületet is, ahol egy barátságos menüsoron keresztül lehet létrehozni VM-eket, valamint egy API-t, amit mind konzolon keresztül (cli-ből), mind külső alkalmazással lehet vezérelni.

Ahhoz, hogy egy VM-et fel tudjunk telepíteni, fel kell tölteni a Proxmox tárhelyére a VM telepítő képet (VM ISO).

Egy VM telepítő konfigurálása általában hosszú folyamat, melyet nagyobb mennyiségű VM előállításánál az üzemeltető nem feltétlen szeretne kivárni, egyesével elvégezni.

CloudInit

Az iparban (valamint a tanszéken is) a legtöbb esetben Ubuntu VM-eket használnak. Létezik egy CloudInitnek nevezett megoldás Ubuntu VM-ek egyszerűbb telepítésére. A CloudInit lehetőséget biztosít arra, hogy előre (a VM indítása és telepítése előtt) beállítsuk a teljesség igénye nélkül az alábbiakat a virtuális gépen:

- felhasználó (létezik egy alapértelmezett, ez az ubuntu felhasználó)
- jelszó (nem kötelező beállítani)
- SSH publikus kulcs(ka)t (ez, ha nem adtunk meg jelszót, a belépéshez elengedhetetlen)
- valamint a hálózati interfészeket lehet konfigurálni:
 - IPv4 és IPv6-os címek
 - IPv4 és IPv6-os hálózati átjárók
 - MAC cím

IaaS (Infrastructure as a Service) megoldás a Proxmox üzemeltetéséhez

Az iparban manapság elterjedt fogalom az Infrastructure as a Service (magyarul infrastruktúra, mint szolgáltatás) – továbbiakban IaaS – rendszerautomatizálás szempontjából kényelmes lehetőségeket biztosít.

Általános megoldás a Terrible kombináció, mely az Ansible és a Terraform eszközök együttes használatából ered.

Az Ansible egy olyan eszköz, melyben deklarativan le tudjuk írni az operációs rendszerünk és a telepített alkalmazásaink kívánt állapotát.

A terraform pedig olyan eszköz, mely segítségével deklarativ konfigurációs nyelven tudnak a fejlesztők definiálni virtuális gépeket (és konténereket). Segítségével gyorsan, öndokumentálóan és könnyen lehet létrehozni a kívánt virtuális gépeket.

Ansible-ös CloudInit

A félév során elkészítettem egy Ansible playbookot (forgatókönyvet), mely segítségével automatikusan fel tudok telepíteni tetszőleges CloudInit-es VM sablont (Vm template) a hosztra, melyeken előtte beállítom a QEMU Guest Agent-öt, mely egy, a vendéggépen futó processz, ami a hoszt OS felé reportol adatokat és segíti a vendég gép vezérlését (pl. leállítás, újraindítás). (*How Ansible Works*)

Terraform-os VM és LXC konténer generálás

Miután elkészítettem a CloudInit-es VM sablonokat (Ubuntu 20.04, 22.04), készítettem két Terraformos konfigurációt is, melyek segítségével VM-eket és LXC konténereket lehet bármekkora mennyiségben, automatizáltan létrehozni. (*What is Terraform?*)

Későbbi fejlesztési lehetőségek

Többi server újratelepítése

A későbbiekben szeretném a maradék három szervert is újratelepíteni Proxmoxal és a Proxmox által használt klaszterizációs megoldást beállítani, hogy a szerverek egy közös Proxmox klaszterként funkcionáljanak. Szeretnék közöttük elosztott fájlrendszer megoldást beállítani, hogy az adatok biztonságosan legyenek tárolva, erre a Ceph-et néztem ki megoldásnak.

GitLab integráció

Szeretnék elkészíteni olyan repozitóri(ka)t, ahol a Terraformas megoldás teljesen ki lenne dolgozva, és biztonságos módon lehetne automatikusan generálni a VM-eket és LXC konténereket.

Kubernetes(ek)

Szeretnék egy, vagy több Kubernetes klasztert létrehozni a gépeken, melyeken a tanszéken dolgozók tudnák a kutatásaikat folytatni. Ehhez szeretnék szintén automatizálható konfigurációt elkészíteni, hogy ha szükség lenne egy újabb Kubernetesre, vagy ha egy meglevőt kellene eldobni és helyette újat létrehozni, akkor azt kényelmesen lehessen elvégezni bárki által.

Összefoglalás

A félév során megismertem a tanszéki szervereket és azok felhasználásának módját. Utána néztem, hogy milyen megoldásokkal lehetne javítani, jobbra tenni a jelenlegi helyzetet, majd telepítettem egy Proxmox hypervisort az egyik tanszéki szerverre.

Elkészítettem egy Ansible-ös és két Terraformas konfigurációt a rendszer automatizálásának előkészítéséhez.

Irodalomjegyzék

Dr. Korn, András. ZFS. <https://unixlinux.tmit.bme.hu/ZFS>. Elérés 2022. május 10.

How Ansible Works. <https://www.ansible.com/overview/how-ansible-works?hsLang=en-us>. Elérés 2022. május 10.

Simic, Sofija. *What is a Hypervisor? Types of Hypervisors 1 & 2*. <https://phoenixnap.com/kb/what-is-hypervisor-type-1-2>. Elérés 2022. május 10.

What are IT mode, HBA mode, RAID mode in (SAS) Controllers? 2021-09-22, <https://dannyyda.com/2021/09/22/what-are-it-mode-hba-mode-raid-mode-in-sas-controllers/>.

What is Terraform? <https://www.terraform.io/intro>. Elérés 2022. május 10.