UPRAVLJANJE VMWARE VIRTUALNIM OBLAKOM KORIŠTENJEM APLIKATIVNOG SUČELJA

Krevzelj, Tomislav

VISOKO UČILIŠTE ALGEBRA

ZAVRŠNI RAD

UPRAVLJANJE VMWARE PRIVATNIM OBLAKOM KORIŠTENJEM APLIKATIVNOG SUČELJA

Tomislav Krevzelj

Zagreb, ožujak 2023.

„Pod punom odgovornošću pismeno potvrđujem da je ovo moj autorski rad čiji niti jedan dio nije nastao kopiranjem ili plagiranjem tuđeg sadržaja. Prilikom izrade rada koristio sam tuđe materijale navedene u popisu literature, ali nisam kopirao niti jedan njihov dio, osim citata za koje sam naveo autora i izvor, te ih jasno označio znakovima navodnika. U slučaju da se u bilo kojem trenutku dokaže suprotno, spreman sam snositi sve posljedice uključivo i poništenje javne isprave stečene dijelom i na temelju ovoga rada“.

U Zagrebu, ožujak 2023.

Predgovor

Ponajviše bih se zahvalo svom profesoru i mentoru mag.ing.el. Vedranu Dakiću koji mi je pružio brojne stručne savjete, strpljenje i pomoć tokom studiranja i pisanja ovog rada. Zahvalio bih se prijateljima i kolegama te obitelji na konstantnoj podršci te vjeri da je ovaj rad moguće privesti kraju. Uz njih, zahvalio bih se i ostalim nastavnicima i djelatnicima Visokog Učilišta Algebra na ugodnom iskustvu studiranja te prilici da usvojim novo i proširim postojeće znanje.

A Ilica 242, 10000 Zagreb
(01) 2222 182
E student@algebra.hr
www.algebra.hr

ALGEBRA

Temeljem članka 8. Pravilnika o završnom radu i završnom ispitu na preddiplomskom studiju Visokog učilišta Algebra sačinjena je ova

Potvrda o dodjeli završnog rada

kojom se potvrđuje da student Tomislav Krevzelj, JMBAG 0246056354, OIB 38143105329 u šk. godini 2021./2022., studij: Primjenjeno računarstvo - Preddiplomski studij, smjer: Sistemsko inženjerstvo, od strane povjerenstva za provedbu završnog ispita, dana 23.03.2022. godine, ima odobrenu izradu završnog rada

s temom: **Upravljanje VMware virtualnim oblakom korištenjem aplikativnog sučelja**

i sažetkom rada: Ovaj rad bavi se izradom Python aplikacije za upravljanje VMware virtualnom okolinom. Primarni fokus stavljen je na konfiguraciju i automatizaciju izrade virtualnih mašina, dok se pritom prolaze i sve popratne radnje kako bi ista bila moguća. Na početku čitatelja upoznajemo s osnovnim konceptima virtualizacije te potom ga uvodimo u svijet VMware-a i pripremamo za izradu vlastite virtualne infrastrukture. U prvom dijelu bavimo se izradom virtualnih mašina putem grafičkog sučelja, dok se u drugom dijelu fokusiramo na izradu i konfiguraciju korištenjem Python aplikacije. Čitatelj će, uz samu izradu virtualnu mašinu, naučiti kreirati korisnike i grupe te definirati pravo pristupa; naučiti kreirati snimke, predloške i kopije virtualnih mašina; proći kroz izradu virtualnih mreža te rekonfigurirati postojeće objekte. U konačnici objedinjujemo postojeće skripte za potpuno automatizirano kreiranje korisnika i virtualne mašine sa željenim parametrima.

Mentor je: Vedran Dakić.

Odobrenjem završnog rada studentu je omogućen upis kolegija "Izrada završnog projekta/Praksa" te je sukladno članku 8. Pravilnika o završnom radu i završnom ispitu dužan najkasnije do početka nastave ljetnog semestra u sljedećoj školskoj godini, uspješno obraniti završni rad uspješnim polaganjem završnog ispita.

U protivnom student može zatražiti novog mentora/icu i temu te ponovo upisati kolegij "Izrada završnog projekta/Praksa" budući da rad koji nije predan i obranjen na završnom ispitu u roku određenom Pravilnikom završnom radu i završnom ispitu prestaje vrijediti. Izrada novog završnog rada se izvodi sukladno rokovima određenima za školsku godinu u kojoj je studentu određen novi mentor/ica i dodijeljen novi završni rad.

Potpis studenta:	Potpis mentora:	Potpis predsjednika povjerenstva:

Ova potvrda izdaje se u 4 (četiri) primjerka od kojih 3 (tri) idu kao prilog završnom radu.

Sažetak

U ovom radu dotaknuti ćemo se teme virtualizacije računalnih resursa na lokalnoj infrastrukturi baziranoj na VMware tehnologiji. Opisati ćemo standardne oblike virtualizacije i njihovu primjenu te prikazati na koji način kreirati virtualne mašine unutar lokalnog okruženja. Potom, uz pomoć Python aplikacije automatizirati ćemo dohvaćanje korisnika koji se nalaze unutar Windows Active Directory imeničkog sustava te izradu izolirane virtualne okoline u kojoj se nalazi virtualna mašina sa željenim parametrima.

Ključne riječi: Virtualizacija, hipervizor, VMware, virtualna mašina, Python, Windows Active Directory, automatizacija.

Summary

This paper will explain the basic steps to achieve computer virtualization based on VMware technology. It will explain standard types of virtualization and their application, as well as how to create a virtual machine. Additionally, with the help of a Python application, we will automate user retrieval from Microsoft Active Directory and the creation of an isolated virtual environment with a virtual machine for each of them.

Keywords: Virtualization, hypervisor, VMware, virtual machine, Python, Windows Active Directory, automatization.

Sadržaj

1. Uvod

Izrada Python aplikacije za upravljanje privatnim oblakom prvenstveno zahtjeva poznavanje virtualizacije i virtualizacijskih tehnologija, stoga se u prvom dijelu rada bavimo upoznavanjem osnovnih koncepata virtualizacije te kako se ona provodi na modernim računalima. Prolazimo kroz tipove virtualizacije, prednosti i mane te upoznajemo čitatelja sa tehnologijom koju ćemo koristiti za izradu virtualnih mašina.

Izradi virtualnih mašina pristupamo metodički, objašnjavajući korake koji vode do željene konfiguracije, te se iz tog razloga u početku bavimo pripremanjem okoline, grafičkim prikazom topologije te konfiguracijom virtualne okoline korištenjem grafičkog sučelja. Potom ulazimo u izradu i konfiguraciju virtualnih mašina, prolazimo kroz korake za izradu predložaka te opisujemo kako od predloška dobiti nove virtualne mašine. Novoizrađene virtualne mašine ćemo potom dodijeliti korisnicima te im omogućiti pristup isključivo do odabranog skupa funkcija unutar vSphere sučelja. Za dodatnu razinu sigurnosti, kreirati ćemo izolirane mrežne segmente unutar kojih ćemo smjestiti virtualne mašine.

Naposlijetku, cijeli proces izrade virtualnih mašina ćemo automatizirati kroz Python aplikaciju, gdje pojedinačno objašnjavamo uvoz korisnika iz imeničkog sustava *Windows Active Directory,* promjenu konfiguracije virtualne mašine i izradu snimki korištenjem API[1] poziva.

S obzirom da je u pitanju složena virtualna okolina, od čitatelja se očekuje napredno poznavanje rada računala, opća snalažljivost te konzultiranje s dodatnom vanjskom literaturom za postizanje istih funkcionalnosti definiranih u ovom radu.

[1] API = Application Programming Interface

2. Virtualizacijska tehnologija

Korištenjem virtualizacijske tehnologije imamo mogućnost pokretanja više virtualnih računala unutar jednog fizičkog računala. Virtualizacija je proces pokretanja virtualnih mašina (instanci) u kojem se apstrahiranjem operacijskog sustava, aplikacije te računalnih mreža iste čine kao uobičajeno fizičko računalo. Te virtualne mašine možemo opisati i kao računalo unutar samog računala, a zahvaljujući činjenici da se spremaju u jednu ili više datoteka, lako ih možemo seliti s jednog računala na drugo.

Virtualizacijom računala možemo učinkovitije iskoristiti resurse koji se na njemu nalaze, a za primjer možemo iskoristiti i klasično računalo sa Intel i7 procesorom te 16GB[2] radne memorije. Za pokretanje aplikacije poput MS Word efektivno će koristiti 1% raspoložive procesorske snage dok bi preostalih 99% čekalo na novi zadatak. Takvo računalo u pravilu ima dovoljno snage da može istovremeno čitati bazu podataka, opsluživati višestruke korisnike sa dijeljenim mapama, glumiti web server te služiti za igranje igrica bez da se sustav preoptereti. Pravilno konfiguriranom virtualizacijom te resurse možemo podijeliti na više virtualnih računala kako bi svako imalo onoliko resursa koliko je potrebno za normalan rad aplikacije koja se na tom računalu nalazi.

Osim bolje iskorištenosti resursa, uvelike poboljšavamo i sigurnost sustava. Kompromitacijom podataka za prijavu u određenu aplikaciju ili operacijski sustav napadač ima pristup isključivo toj aplikaciji, odnosno tom operacijskom sustavu. S obzirom da svako virtualno računalo možemo zamisliti kao zaseban logički segment, time uvelike umanjujemo doseg te štetu koju napadač može uzrokovati unutar sustava. Korištenjem višestrukih slojeva zaštite lakše je otkriti maliciozne, ali i spriječiti legitimne korisnike u pokušaju pristupa sadržaju koji im nije namijenjen.

Uz pomoć virtualnog okruženja moguće je kreirati zasebne mrežne segmente te i na taj način odijeliti jedan dio sustava od drugog čak i ako se oba nalazila na jednom fizičkom računalu. Na taj način možemo odvojiti produkcijsku okolinu od testne, ali i iskoristiti virtualno računalo kao ulaznu točku za virtualnu privatnu mrežu (*eng. Virtual Private Network - VPN*).

[2] GB = Gigabajt

U tom slučaju se korisnike koji pristupaju izvan sustava smješta u zaseban mrežni segment bez potrebe da se investira u dodatni hardver (poput vatrozida nove generacije, *eng. Next-Generation Firewall - NGFW*) koji se inače koristi u tu svrhu.

Kreiranjem virtualnog okruženja možemo testirati ponašanje softvera na više različitih operacijskih sustava (Windows ili Linux), pa čak i na više verzija operacijskog sustava. Softver koji ispravno radi na originalnoj Windows 10 verziji možda neće ispravno raditi i na verziji Windows 10 20H2 – u tu svrhu možemo kreirati virtualno računalo kako bi utvrdili funkcionalnost aplikacije.

Tehnologiju virtualizacije računalnih resursa primarno koristimo na specijaliziranom hardveru u poslovnom okruženju, gdje je investiciju u hardver i softver računala potrebno maksimalno iskoristiti. Ista se može realizirati i korištenjem hardvera namijenjenom standardnim korisnicima kako bi se pokrenuli različiti operacijski sustavi, kreirala testna okolina, napravila sigurnosna kopija (*eng. Backup*) računala i sl. U pravilu, za virtualizaciju hardvera koji nije namijenjen za korištenje u serveru nema direktne podrške proizvođača, no to ne znači da se upute ne mogu naći na brojim forumima te IT[3] portalima.

Ovisno o veličini virtualne infrastrukture, kompanije će se najčešće odlučiti za neku od 5 vodećih virtualizacijskih platformi[4]:

- VMware
  - Vodeći u robusnosti i sigurnosti te sa širokom lepezom proizvoda za upravljanje i nadziranje virtualnih okolina u lokalnom okruženju. Predstavljanjem VMware Cloud Foundation omogućili su jednostavnu međusobnu integraciju svojih alata za hibridne instalacije virtualnih okolina.
- Amazon
  - Amazon Web Services (AWS) je vodeća i najstarija platforma u oblaku sa najvećim brojem funkcionalnosti i opcijama za mikromenadžment pojedinih usluga i servisa.
- Microsoft
  - Nudi uslugu virtualizacije u oblaku putem Microsoft Azure platforme ili lokalno korištenjem Hyper-V hipervizora. Po zastupljenosti i funkcionalnosti su odmah iza AWS-a[5]

[3] IT = informacijska tehnologija
[4] Izvor: https://www.educba.com/virtualization-platforms
[5] Vrijednosti do drugog kvartala 2022. god, Izvor: https://www.wpoven.com/blog/cloud-market-share

- Google
  - Iako je Google Cloud najnovija platforma naspram Microsofta i Amazona te sa funkcionalnosti zaostaje za svojim rivalima, ima jednu od najbrže rastućih platformi za usluge u oblaku[6]
- Oracle
  - Kao i u slučaju VMware, nude lokalnu virtualizaciju sa alatima VirtualBox i Oracle Linux Virtualization Manager, te uslugu softvera kao servisa (eng. Software as a Service – SaaS) za pokretanje njihovih aplikacija u oblaku.

2.1. Tipovi virtualizacije i usluga u oblaku

Virtualizaciju možemo podijeliti u nekoliko kategorija, a 5 najčešćih tipova virtualizacije su[7]:

- Virtualizacija aplikacija (*eng. Application Virtualization*)
- Virtualizacija mreže (*eng. Network Virtualization*)
- Virtualizacija servera (*eng. Server Virtualization*)
- Virtualizacija pohrane (*eng. Storage Virtualization*)
- Virtualizacija radne površine (*eng. Desktop Virtualization*)

Virtualizacija aplikacije omogućava korisnicima pristup aplikaciji koja se nalazi na serveru. Aplikacija se pritom pokreće u zasebnom logičkom okruženju (eng. sandbox), gdje se resursi potrebni za rad aplikacije emuliraju dok se aplikaciji čini da radi na klasičnom operacijskom sustavu.

Virtualizacija mreže podrazumijeva kreiranje apstraktnog sloja mreže koji je odvojen od fizičkog hardvera. Postavke poput IP adresa, VLAN segmenata te pravila za vatrozid mogu biti primijenjena na virtualno sučelje, bez potrebe da se za to koriste fizički ulazi na usmjerniku ili preklopniku. Tim pristup možemo kvalitetnije segmentirati mrežu te stvoriti uvjete za testiranje ispravnosti aplikacija, ali i lakši nadzor prometa.

Virtualizacija poslužitelja odnosi se na podjelu resursa koji su dostupni unutar servera na više virtualnih instanci. Potom te virtualne instance, uz pomoć hipervizora, pristupaju resursima koje se nalaze na serveru - poput procesora, memorije te pohrane.

[6] Izvor: https://intellipaat.com/blog/aws-vs-azure-vs-google-cloud/
[7] Izvor: https://www.geeksforgeeks.org/virtualization-cloud-computing-types/

Virtualizacija pohrane jedan je od ključnih dijelova za uspješno poslovanje za većinu manjih, a naročito većih firmi. Klasični korisnik virtualnu pohranu može zamisliti kao mrežni disk koji se nalazi na udaljenom mrežnom uređaju (eng. NAS – Network Attached Storage). Sa porastom potrebe za pohranom podataka, te instalacijom više servera za pohranu, potrebno je voditi i evidenciju kome to podaci pripadaju te gdje se nalaze. Softver za virtualizaciju pohrane radi upravo to te regulira tko, kako i koliko često netko ima pristup toj pohrani.

Virtualna radna površina je u osnovi virtualna mašina koja se najčešće nalazi na udaljenom serveru unutar kompanije. Virtualnom radnom površinom moguće je uštedjeti na hardveru i softveru koji su potrebni za rad zaposlenika, lakše i ujednačenije ažurirati sustav i aplikacije na novije verzije, ograničiti pristup i dijeljenje povjerljivih datoteka te standardizirati i automatizirati ispostavu novih radnih površina za buduće korisnike.

Unutar okruženja u oblaku najčešće ih sortiramo u 3 kategorije:

- Softver kao servis (*eng. Software-as-a-Service - SaaS*)
- Platforma kao servis (*eng. Platform-as-a-Service - PaaS*)
- Infrastruktura kao servis (*eng. Infrastructure-as-a-Service - IaaS*)

Uz gore navedene, ponekad ćemo naići na druge aaS-ove poput:

- *DaaS* gdje „D“ označava Desktop, odnosno radnu površinu
- *DBaaS* gdje se „DB“ odnosi na Database, odnosno bazu podataka,
- *CaaS* gdje je C oznaka za kontejner,

a „aaS“ uvijek označava „kao servis“. Konačni cilj je ponuditi funkcionalnu uslugu gdje administratori imaju minimalnu količinu konfiguracije i administracije za postizanje funkcionalne usluge koju potom isporučuju klijentima. Usluge u oblaku omogućavaju firmama u razvoju da u startu koriste najmodernije i visoko dostupne tehnologije bez ulaganja u skupocjeni hardver i softver računala jer se kompletna usluga licencira od strane davatelja usluge i naplaćuje po satu rada. Kombinacijom lokalnog i okruženja u oblaku možemo izvući najbolje od obje strane – čuvati povjerljive financijske podatke unutar firme, a pokretanje web stranice povjeriti davatelju usluge kako bi osigurali maksimalnu dostupnost za krajnje korisnike.

Uz sve gore na umu, možemo prikazati komparacijsku tablicu odgovornosti administratora ovisno o odabranom tipu virtualizacije resursa:

Tablica 2.1 Odgovornost administratora ovisno o odabranoj virtualizaciji resursa

Lokalno smješteni server *(On-Premise)*	Infrastruktura kao servis (*IaaS*)	Platforma kao servis (PaaS)	Softver kao servis (SaaS)
Aplikacije	Aplikacije	Aplikacije	Aplikacije
Podaci	Podaci	Podaci	Podaci
Servisi	Servisi	Servisi	Servisi
Operacijski sustav	Operacijski sustav	Operacijski sustav	Operacijski sustav
Virtualizacijska tehnologija	Virtualizacijska tehnologija	Virtualizacijska tehnologija	Virtualizacijska tehnologija
Serveri	Serveri	Serveri	Serveri
Pohrana	Pohrana	Pohrana	Pohrana
Mreža	Mreža	Mreža	Mreža

Održava administrator (korisnik) usluge

Održava davatelj usluge u oblaku

Odavde je lako zaključiti da najveću odgovornost preuzimamo prilikom pokretanja virtualizacije na vlastitoj infrastrukturi, dok najmanju prilikom korištenja softvera kao servisa – u tom slučaju brinemo isključivo o ispravnoj konfiguraciji tog servisa.

2.2. Javni, privatni i hibridni oblak

Javnim oblakom nazivamo svu infrastrukturu koja je dostupna široj populaciji, te gdje se resursi mogu zakupiti ovisno o potrebi organizacije. Iako je moguće zakupiti fizičke resurse, javni oblak se u pravilu koristi kao dijeljena infrastruktura, odnosno, jedan server opslužuje više korisnika. Na taj način se postiže veća efikasnost iskorištenih resursa, a korisnik plaća samo onoliko koliko mu i treba. Također, javni oblaci u potpunosti rješavaju brigu korisnika oko hardvera računala. Eliminiraju brige oko opskrbom električne energije ili dostupnošću internetske veze. Zbog velike propusnosti podataka u internetskom prometu te mogućnosti odabira različitih geografskih lokacija za smještaj podataka, čest su odabir za čuvanje sigurnosnih kopija i opsluživanje web stranica s minimalnom latencijom do krajnjih

korisnika. Naravno, za korištenje javnih oblaka korisnici se moraju složiti s određenim pravilima o korištenju usluga, gdje se davatelji usluga u oblaku najčešće u potpunosti ograde od mogućeg gubitka podataka zbog fizičkih ili softverskih kvarova. Korisnik je isto tako dužan brinuti o sigurnosti vlastitih podataka te voditi brigu o tome da se na javni oblak ne spremaju maliciozne datoteke.

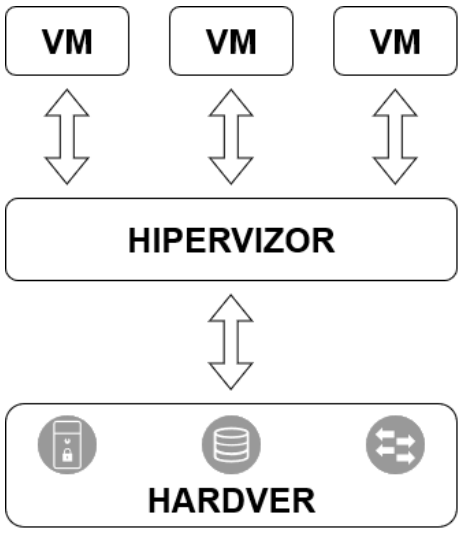
Privatni oblaci nude veću kontrolu nad podacima i osjećaj sigurnosti jer se serveri obično nalaze unutar same kompanije, te se tim oblakom koristi jedna organizacija. Za pokretanje privatnog oblaka potrebne su daleko veće investicije u startu te redovita održavanja i nadogradnje sustava. Kako se serveri nalaze lokalno, administratori trebaju brinuti o fizičkoj i logičkoj zaštiti servera i aplikacija, te implementirati vlastite alate za nadgledanje sustava. Zbog svoje niske latencije, privatni oblak je odličan izbor za interne aplikacije i servise kojima se redovito pristupa te koje moraju biti dostupne čak i u slučaju gubitka internetske veze. Ovisno o vrsti informacija koje se na servere spremaju, privatni oblak može biti i jedino rješenje koje će udovoljiti regulativama i ostalim zakonskim odredbama koje definiraju način postupanja s takvim podacima.

Hibridni oblak je spoj privatnog i javnog oblaka. Oni koji ne spada u kategoriju vladinih organizacija te mogu podatke pohranjivati i u drugim državama, najčešće odabiru najbolje od oba svijeta. Smještanjem i odvajanjem aplikacija koje su dostupne krajnjim korisnicima te onih koje su namijenjene interno zaposlenicima može uvelike povećati sigurnost podataka, omogućiti rad čak i za vrijeme održavanja lokalnih servera te kreiranje sigurnosnih kopija na više fizičkih lokacija za lakši opravak u slučaju katastrofe.

Gore navedena rješenja je moguće kombinirati i spajati u multi oblak rješenja. Na taj način možemo povezati dvije lokacije na kojima se nalazi privatni oblak te određene servise i aplikacije pokretati na jednoj, odnosno drugoj lokaciji. Ista stvar može se postići kombinacijom javnih oblaka na način da npr. kontejnerizirane aplikacije izvode djelomično na Google Cloud platformi, a djelomično na Microsoft Azure platformi. Osim iz sigurnosnih razloga, na taj se odabir administratori odluče i zbog činjenice da im je lakše i jeftinije pokrenuti i administrirati određenu uslugu, a ponekad zbog činjenice da neke funkcije jednostavno nisu dostupne kod ostalih davatelja usluge u oblaku.

2.3. Hipervizor

Hipervizor je softver koji služi za odvajanje fizičkih resursa koji su dostupni na računalu od virtualne okoline. U osnovi, služi kao upravljački sloj preko kojeg se odvija sva komunikacija između fizičke i virtualne okoline. Hipervizori se mogu nalaziti unutar operacijskog sustava (npr. Microsoft Hyper-V) ili se instalirati bez OS[8]-a na hardver računala (npr. VMware ESXi). Pojmovi kojima vežemo uz te vrste instalacije su Hosted Virtualization te Bare-metal Virtualization. U praksi se preferira virtualizacija koja se ne oslanja na operacijski sustav (Bare-Metal) zbog činjenice da svakim dodatnim slojem u virtualizaciji ujedno povećavamo i latenciju pristupa resursima, kao i samu ranjivost sustava. Zbog toga što hipervizor služi kao posrednik u komunikaciji virtualnih mašina sa hardverom, unutar hipervizora možemo adekvatno raspodijeliti te ujedno i ograničiti pristup hardverskim resursima koji se nalaze na serveru. Na taj način postižemo bolju ukupnu iskoristivost servera jer se svakoj virtualnoj mašini dodjeljuje onoliko resursa koliko je potrebno za rad aplikacija koje se na njoj nalaze.



Slika 2.1 Komunikacija virtualnih mašina s hardverom putem hipervizora

Većina hipervizora programirana je u vidu skalabilnosti, s obzirom da današnje potrebe za resursima nadilaze i one najjače konfiguracije servera. Velike baze podataka poput Google

[8] OS = Operacijski sustav

Gmail elektroničke pošte, komentara na Facebook društvenoj mreži ili rezultati simulacija unutar CERN institucije za nuklearna istraživanja jednostavno ne stanu čak ni na najsnažnija računala na svijetu. Pojedine aplikacije na virtualnim mašinama ponekad moraju biti dostupne 24 sata u danu te 365 dana u godini. Dovoljno je zamisliti što bi se dogodilo kada bi se server na kojem se nalazi aplikacija za održavanje pacijenta na životu iznenada ponovo pokrenuo ili u potpunosti prestao raditi. Iz tog su razloga razvijeni razni alati koji imaju mogućnost grupiranja i praćenja rada servera, sigurnost operacijskog sustava te stabilnost aplikacija koje se na njemu izvode.

Hipervizore je moguće grupirati dodavanjem više fizičkih servera u tzv. klaster (*eng. Cluster*). Klaster u ovom slučaju predstavlja grupu servera koji dijele komputacijsku snagu za pokretanje virtualnih mašina. Ako zamislimo 5 servera u klasteru od kojih svaki ima po 10GHz procesorske snage, 32GB radne memorije (RAM) te 1TB[9] diskovnog prostora, u klasteru će se prikazati kao 50GHz procesorske snage, 160GB radne memorije te 5TB diskovnog prostora. Uz uvjet da na raspolaganju imamo dovoljnu brzu međusobnu komunikaciju između servera, te resurse potom možemo raspodijeliti ovisno o potrebama virtualnih mašina. S obzirom da hipervizor sam po sebi nema mogućnost kreiranja klaster, u tu svrhu koristimo alat (softver) koji zna komunicirati i upravljati s hipervizorom poput VMware vCenter Server ili System Center Virtual Machine Manager (SCVMM).

2.4. Prednosti i mane virtualizacije

Za ljude koji rade u IT-u, virtualizacija računalnih resursa ujedno može biti izvor i tuge i radosti. U jednu ruku, virtualizacija omogućava velike uštede novaca, a u drugu, bez pravilnog postavljanja i održavanja može izazvati neviđene glavobolje, posebno dok se zbog sitnica potroše sati da bi se došlo do izvora problema. Zato ćemo sumarizirati neke od najvažnijih prednosti i mana koji se vežu uz virtualizaciju sustava.

[9] TB = Terabajt

Prednosti virtualizacije:

- Visoka iskoristivost kompjuterskih resursa
  - Ono što je najočitije i na što se prvotno cilja je maksimizirati investiciju u hardver. Idealna virtualizacija je ona koji u praksi konstantno koristi 100% iskoristivih resursa, dok se u praksi cilja da taj broj bude između 80-90%[10] kako se sustav ne bi nepotrebno usporio. Virtualizacija nam omogućava i tzv. *Overprovisioning* gdje npr. 2 jezgre možemo dodijeliti na više virtualnih mašina gdje će svaka imati po 2 jezgre. Razlog tome je što u većini slučajeva procesor ne radi na 100% opterećenosti te tu razliku imamo priliku iskoristiti na drugoj virtualnoj mašini. Isto vrijedi i za radnu memoriju i diskovni prostor.
- Skalabilnost
  - Druga najbitnija stavka prilikom odluke za uvođenje virtualizacije je jednostavna skalabilnost. U trenu kada ponestane resursa, potrebno je dodati novi server u klaster ili, ako to budžet ne dozvoljava, nadograditi stari. Server računala su po prirodi zamišljena na način da ih je jednostavno održavati, zamjena komponenti traje svega par minuta, a za određene nadogradnje čak nije potrebno ni gasiti server.

- Visoka dostupnost
  - Server računala su zamišljena da rade 24h u danu te 365 dana u godini. Za pristup aplikacijama ne treba paliti kolegino računalo niti će padom vlastitog nestati podaci na kojima smo radili. Upotrebom više servera, kritične aplikacije možemo smjestiti na nekoliko virtualnih mašina te će one uvijek biti dostupne čak i ako je neki od servera potrebno servisirati.
- Oporavak od katastrofe
  - Virtualne mašine lakše je sigurnosno kopirati i spremiti na izvanmrežnu lokaciju. Ponekad je potrebno upotrijebiti redoslijed sigurnosnog kopiranja i oporavka sustava jer npr. web aplikacija ne može raditi bez baze podataka. Aplikacija i baza se mogu nalaziti na dvije različite virtualne mašine te je iz tog razloga bitno da se prvo osposobi baza kako bi aplikacija ispravno radila po oporavku. Naravno, tu nema potrebe za paljenjem i gašenjem niti trčanjem od jednog računala do drugog jer je sve virtualno te se sve može odraditi s jedne lokacije.
- Migracija sustava
  - Svaka virtualna mašina spremljena je na disk u obliku jedne ili više datoteka. Da bi u potpunosti premjestili virtualnu mašinu sa operacijskim sustavom, datotekama i aplikacijama koje su na njoj, jednostavno je potrebno premjestiti te datoteke na novi server. Taj server može biti u oblaku ili lokalno što je isto jedan od benefita prilikom kreiranja hibridne okoline.
- Jednostavno kreiranje novih virtualnih mašina
  - Nakon inicijalnog postavljanja i kreiranja predložaka za virtualne mašine, nove se mogu instalirat i pokrenut u svega par minuta

[10] Izvor: Ken Leoni, https://www.heroix.com/blog/vmware-vcpu-over-allocation/

- Precizan uvid u potrošnju resursa
  - Hipervizori konstantno nadziru količinu zauzetih resursa na serveru te je moguće dobiti precizan uvid kada i koliko virtualne mašina pristupaju procesoru, radnoj memoriji, diskovima te mreži te ih potom grafički prikazati za lakše iščitavanje.
- Štednja energije i prostora
  - Prebacivanjem sa fizičke na virtualnu okolinu štedimo prostor i energiju. Uz manju potrošnju energije ujedno i manje zagrijavamo prostor te su sukladno tome inicijalne, ali i buduće investicije u rashladni sustav niže. Uz sve to, buku koju proizvode serveri možemo izolirati u jednoj prostoriji te onemogućiti pristup neovlaštenim osobama.

Mane virtualizacije:

- Visoka inicijalna investicija
  - Virtualizacijski softver najčešće se licencira po broju procesora unutar servera i/ili broju jezgri unutar procesora. Ako želimo imati stručnu podršku, ono najbolje od najboljeg košta. Ti iznosi mogu doseći stotine tisuća kuna samo za virtualizacijski softver, a pogotovo ako dolazi u modularnom izdanju gdje se određene funkcije dodatno naplaćuju. Uz skupi softver bira se i skupi hardver, jer je kudikamo lakše izdvojiti takve novce za 2 procesora u kojem u svakom od njih ima po 64 jezgre, nego za 2 procesora gdje u svakom ima po 4 jezgre. Takve investicije mogu doseći i milijunske iznose što je jedan od primarnih razloga zašto ljudi odustanu od virtualizacijske tehnologije.
- Novi softver
  - Kako bi ispravno konfigurirali virtualizacijsku okolinu, potrebno je znanje oko instalacije i konfiguracije virtualizacijskog softvera, što zahtjeva dodatna vremenska i novčana ulaganja u obuku ljudi koji će njim upravljat. Čak i one besplatne varijante virtualizacije se mogu pokazati izuzeto skupe kada dođe do pada sustava, a dokumentacija ne pokriva taj specifičan slučaj.
- Pad hipervizora
  - S obzirom da je hipervizor softver koji nadzire rad virtualnih mašina, njegovo rušenje će rezultirati time da sve virtualne mašine postanu nedostupne. To nas može dovesti u nezgodnu i naizgled bespomoćnu situaciju ako postoji određeni protokoli za gašenje i ponovo pokretanje tih virtualnih mašina.
- Nadogradnja sustava
  - Hipervizor nije vrsta softvera koja se jednom instalira i potom se zaboravi. Kao i svaki drugi softver, periodički izlaze nadogradnje koje dodaju nove funkcionalnosti, poboljšavaju stabilnost, ali i krpaju ranjivosti koje hakeri mogu iskoristiti za neovlašteni upad u sustav. Iz tog je razloga ponekad potrebno odabrati modularne hipervizore gdje se nadogradnja može obaviti bez potrebe za gašenjem cijelog sustava, iako je i to ponekad neizbježno.
- Problematične performanse
  - Neispravna konfiguracija virtualizacijskog softvera i/ili korištenje nepodržanog hardvera može rezultirati poražavajućim performansama u virtualnoj okolini.

Spori diskovi, mreža, slab procesor i nedostatak radne memorije su samo neki od čimbenika koji mogu usporiti ili totalno onemogućiti pristup i rad na virtualnim mašinama. Isto tako ono što je prednost, ujedno je i mana – dodjeljivanje virtualnih resursa koji nadilaze fizičke kapacitete može rezultirati nefunkcionalnošću i kompletnim padom sustava.

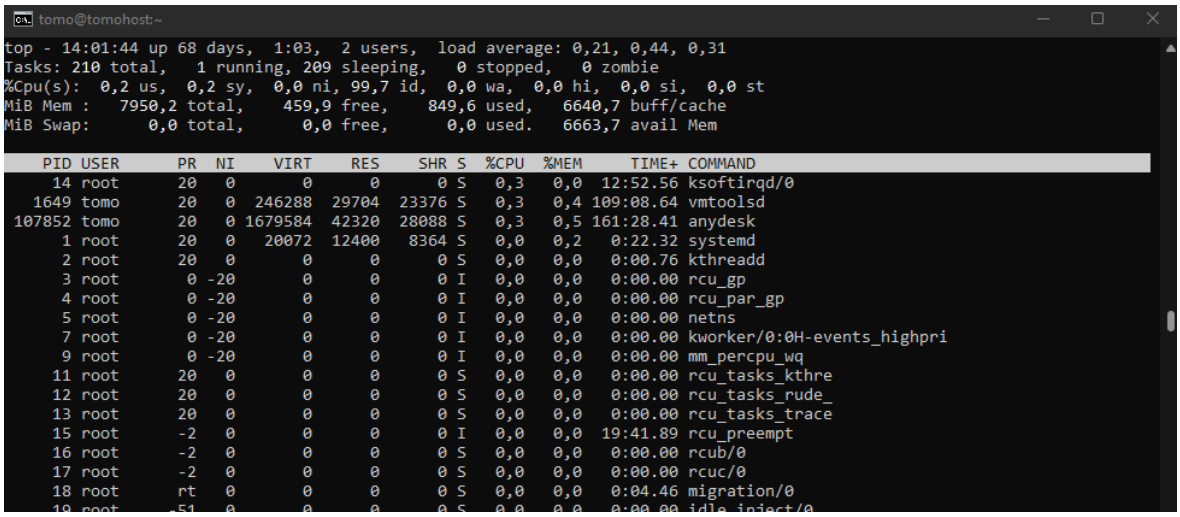
- Sigurnost podataka
  - Iako s pravilnom enkripcijom i konfiguracijom podataka povećavamo sigurnost u virtualnoj okolini, isto tako neispravnom konfiguracijom potencijalno izlažemo te podatke na uvid neovlaštenoj osobi. Tko je zapravo vlasnik podataka i tko sve ima uvid u njih kada se oni nalaze na udaljenoj lokaciji je i dan danas jedna od najškakljivijih tema gdje nema pravog odgovora. Nitko od nas ne želi da treća strana ima uvid u naše osobne i financijske podatke, pogotovo kada se ti podaci nalaze u oblaku.

2.5. Odabir hardvera za virtualizaciju

Prilikom odabira hardvera za server, u vidu moramo imati nekoliko stvari koje značajno mogu utjecati na performanse servera. Za početak, treba krenuti od vrste aplikacija koje će se na tom serveru pokretati. Neka od pitanja koja si trebamo postaviti su:

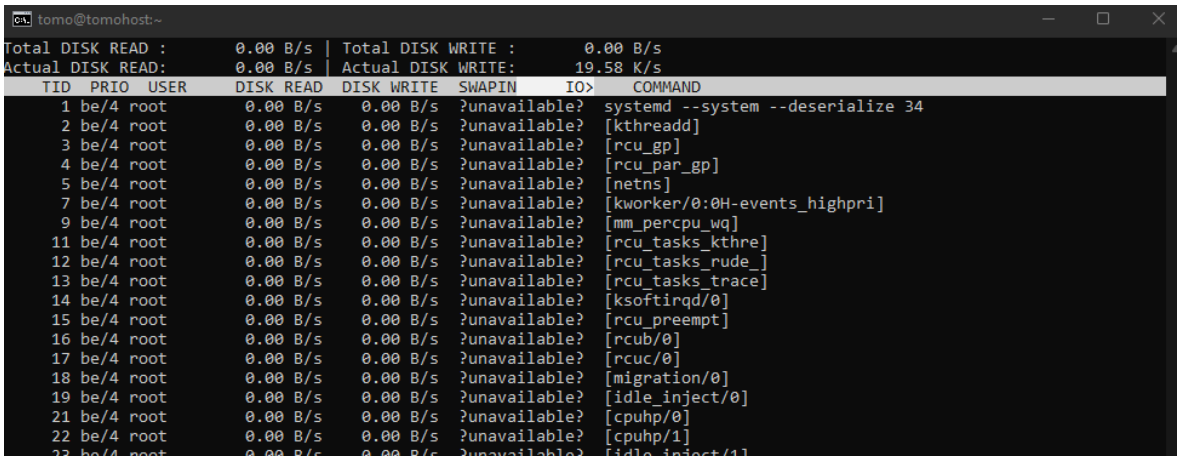
- Koje su to aplikacije i servisi koje želimo virtualizirati?
- Koliko su te aplikacije procesorski i memorijski zahtjevne?
- Na koje se sve servise oslanjaju?
- Koliko često i koju količinu podataka pišu po disku?
- Koliko je bitna latencija pristupa disku?
- Koliko brzu mrežu zahtijevaju?
- Mora li aplikacija biti stalno dostupna?

Komadne poput „top“ i „iotop“ na Linux operativnim sustavima nam mogu dati uvid u zauzeće procesorske i radne memorije, te količinu ulazno-izlaznih operacija po sekundi koje su prisutne na disku (*eng. Input/Output operations per second = IOPS*). Naredba „top“ ispisuje aktivne procese, njihov identifikacijski broj te njihovo zauzeće radne memorije i procesorske snage. Također, moguće je iščitati ukupno zauzeće resursa u sustavu.



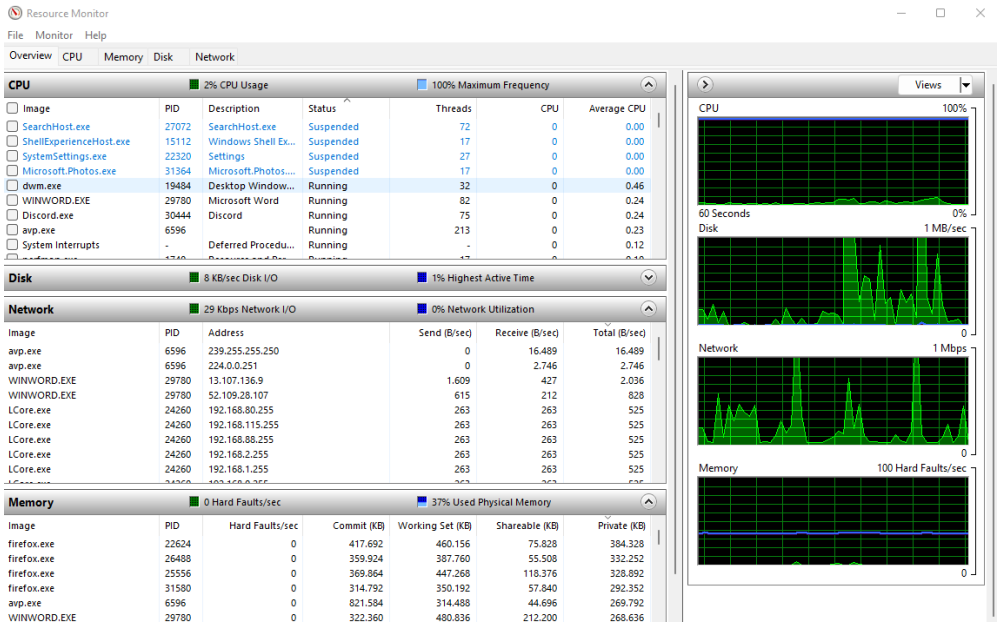
Slika 2.2 Rezultat naredbe *top* na Linux operacijskom sustavu

Korištenjem naredbe „iotop“ dobivamo tablicu unutar koje možemo iščitati trenutno aktivne ulazno-izlazne operacije na disku, raspoređene po procesima na sustavu.



Slika 2.3 Rezultat naredbe *iotop* na Linux operacijskom sustavu

Na Windows operacijskom sustavu slične rezultate možemo vidjeti pomoću alata imenom „Monitor Resursa“ (*eng. Resource Monitor*).



Slika 2.4 Monitor resursa na Windows operacijskom sustavu

Te rezultate potom možemo pratiti u vremenskim intervalima, u vrijeme najvećeg opterećenja te u vrijeme minimalnog opterećenja kako bi dobili okvirne vrijednosti potrebnih resursa za određene aplikacije. Mjerenja se najbolje mogu obaviti na već postojećoj funkcionalnoj okolini, no u slučaju kada to nije moguće, potrebno je procijeniti potrebne resurse prema specifikacijama proizvođača softvera. Dobivene rezultate treba uzeti s dozom opreza jer uz nenadana visoka opterećenja treba ostaviti prostora i za rast aplikacije - opterećenje neće biti isto pri upravljanju sa 1.000 i 1.000.000 zapisa.

Za neke aplikacije će trebate manje od jedne jezgre ili jednog GB radne memorije, što možda neće biti moguće odabrati prilikom izrade virtualne mašine. U tom se slučaju isti resursi mogu podijeliti između nekoliko virtualnih mašina (*eng. overprovisioning*), no takve vrste instalacije treba držati pod budnim okom kako ne bi došlo da zasićenja resursa.

Procesorsku snagu unutar računala možemo izračunati na sljedeći način:

- (Broj procesorskih utora) x (Broj jezgri po procesoru) = (Ukupno fizičkih jezgri)

Za procesore koji podržavaju hipernitnost (eng. hyperthreading):

- (Broj procesorskih utora) x (Broj jezgri po procesoru x 2) = (Ukupno virtualnih jezgri)

Ukupnu snagu u GHz možemo izraziti na sljedeći način:

- (Ukupno fizičkih jezgri) x (Takt procesora) = (Ukupna snaga procesora u GHz)

Primjer izračuna za 2 fizička procesora modela „Intel® Xeon® Processor E5-2695 v3 35M Cache, 2,30 GHz“ bio bi sljedeći:

- Ukupno fizičkih jezgri: 2 x 14 = 28
- Ukupna snaga u GHz: 28 x 2,3 GHz = 64,4 GHz
- Ukupno virtualnih jezgri: 2 x 14 x 2 = 56

Izračun radne memorije vrši se na način da se pomnoži kapacitet memorije sa brojem memorijskih modula - u pravilu koriste memorijski moduli istog kapaciteta iako je moguće koristiti i module različitih kapaciteta. U slučaju da imamo 8 memorijskih modula od kojih svaki ima po 32 GB radne memorije, izračun je sljedeći:

- Ukupna radna memorija: 32 GB x 8 = 256 GB

Diskovni prostor računamo na sličan način kao i radnu memoriju – pomnožimo broj diskova sa njihovim kapacitetom. Za 8 diskova od kojih svaki ima po 960 GB, izračun je sljedeći:

- Ukupni diskovni prostor: 960 GB x 8 = 7.68 TB[11]

U teoriji, ako virtualna mašina zahtjeva u prosjeku zahtjeva 2 GHz procesorske snage, 8 GB radne memorije te 200GB diskovnog prostora, na taj server možemo smjestiti 32 virtualne mašine. U praksi, potrebno je uzeti još brojne čimbenike u obzir prilikom odabira komponenti za server. Stvari poput brzine i tipa radne memorije, broja ekspresno međusobno povezivih perifernih komponenti (*eng. Peripheral Component Interconnect Express = PCIe*), brzine sabirnice, brzina pristupa disku, količina ulazno-izlaznih operacija po sekundi te mnoge druge također igraju veliku ulogu pri odabiru komponenti za server. No, dosad opisani pojmovi biti će dovoljni za interpretaciju grafova i izradu virtualnih mašina čime ćemo se pozabaviti u idućim poglavljima.

[11] TB = Terabajt

3. VMware virtualizacijska platforma

VMware je jedna od najvećih i vodećih kompanija u virtualizacijskoj tehnologiji[12] smještena u Palo Alto u Kaliforniji. Njihov softver omogućava virtualizaciju gotovo svakog dijela računala, a raspoložive resurse prezentira u obliku softversko definiranog podatkovnog centra (*eng. Software-defined data center* - *SDDC*). Softver je namijenjen poduzećima koji traže moderno rješenje za dinamičku konfiguraciju i instalaciju aplikacija, infrastrukture te ostalih IT resursa. Današnje web aplikacije su agilne, svakodnevno se ažuriraju, skaliraju, sele sa jednog servera na drugi te pristupaju i isporučuju sadržaj koji se uopće ne nalazi u lokalnoj bazi. Jedini način za isporučiti usluge i standarde na skali koje današnja tehnologija postavlja je automatizacijom, a jedini način da automatizacija ima smisla je da se dostupni hardverski resursi predstave u obliku softverskih. Softversko definirani podatkovni centar omogućava upravo to, a administratorima ostavlja na izbor na koji način će se ti resursi koristiti. komadića da se Gotovo svaki komad softvera se plaća, iako postoje i besplatne varijante u obliku probne i nekomercijalne uporabe. Softver koji je nekomercijalne uporabe je limitiran funkcionalnošću, a za potrebe izrade virtualne okoline morati ćemo odabrati probnu verziju ili upotrijebiti valjani ključ za komercijalnu uporabu.

3.1. VMware ESXi hipervizor

VMware ESXi je moćni hipervizor koji se primarno koristi u velikim organizacijama kao virtualizacijska platforma za poslovnu okolinu. Omogućava veliku fleksibilnost te pregršt funkcionalnosti putem web alata, od kuda se i upravlja samom platformom.

ESXi, kao i ostale virtualizacijske platforme, omogućava pokretanje više virtualnih mašina na jednom serveru. Moguće je kreirati virtualne mašine sa željenim operacijskim sustavom, brojem procesorskih jezgri, količinom radne memorije te vrstom i količinom mrežnih kartica. Time efektivno omogućava organizacijama koje imaju potrebe za više fizičkih računala sa različitim operacijskim sustavima da ih smjeste u nekoliko virtualnih. Pomoću web alata možemo pratiti funkcionalnost i opterećenje hardverskih komponenti, kao i

[12] Izvor: Gartner Peer Insights, https://www.gartner.com/reviews/market/server-virtualization

virtualnih računala pojedinačno. Takve funkcionalnosti su poželjne prilikom pojave kvara i otklanjanja istog.

Glavna prednost ESXi-a naspram ostalih hipervizora je ta što virtualne mašine ne moraju prolaziti kroz još jedan sloj virtualizacije, kao što je to slučaj s Microsoft Hyper-V hipervizorom. To rezultira manjim latencijama i većom količinom iskoristivih resursa, što možda nije toliko zamjetno kada se govori o procesorskoj snazi i radnoj memoriji, ali dolazi do izražaja kada su u pitanju diskovni IOPS[13]-i. Također, operacijski sustav može ograničiti količinu virtualne memorije, tako da kada želimo upotrijebiti nekoliko grafičkih kartica na operacijskom sustavu Windows, potrebno je napraviti određene izmjene u sustavu, specifično *Pagefile*, kako bi tu memoriju mogli iskoristiti. U slučaju ESXi-a nemamo takvih ograničenja, niti je potrebno mijenjati postavke da bi virtualne mašine mogle iskoristiti resurse koji se na tom serveru nalaze.

VMware ESXi je platforma koja se tokom godina pokazala veoma stabilnom naspram konkurencije. Dok se drugi hipervizori ruše pri neočekivanim padu operacijskog sustava, naglim porastima potražnje za resursima ili jednostavnom smrzavanju hipervizora, ESXi se uspješno snalazi u svim izazovima koji se pred njega postave. U slučaju smrzavanja ili neresponzivnosti, dovoljno je ponovo pokrenuti odgovarajući servis kako bi ga doveli u funkcionalno stanje, pritom neometajući rad virtualnih mašina koje se na njemu nalaze. Zbog toga što je instalacija hipervizora iznimno mala, moguće je instalirati ESXi na USB disk ili memorijsku karticu, pritom ne brinući o životnom vijeku istih s obzirom da je količina pisanja u radu gotovo nepostojeća. Disku se u prosjeku pristupa jednom u sat vremena, za izradu logova, iako ako se u tu svrhu koristi zaseban disk, mogli bi proći i dani prije nego što se išta novo zapiše na disk koji služi za pokretanje sustava[14]. U slučaju kvara diska ili korupcije instalacije, dovoljno je nanovo instalirati ESXi na novi disk te prenijeti konfiguracijsku datoteku – što se sve može odraditi u nekoliko minuta.

Osim konfiguracije samih virtualnih mašina, ESXi možemo iskoristiti i za konzolni pristup, konfiguraciju i instalaciju tih virtualnih mašina. Taj način nam omogućava pristup na virtualnu mašinu čak i kada mreža zakaže, ili se neki od upravljačkih programa sruši, kao i u slučaju da je ispred nas fizičko računalo. Samim ESXi-em se upravlja pojedinačno ili pomoću VMware vCenter servera, gdje se potonje koristi kada imamo nekoliko servera koje

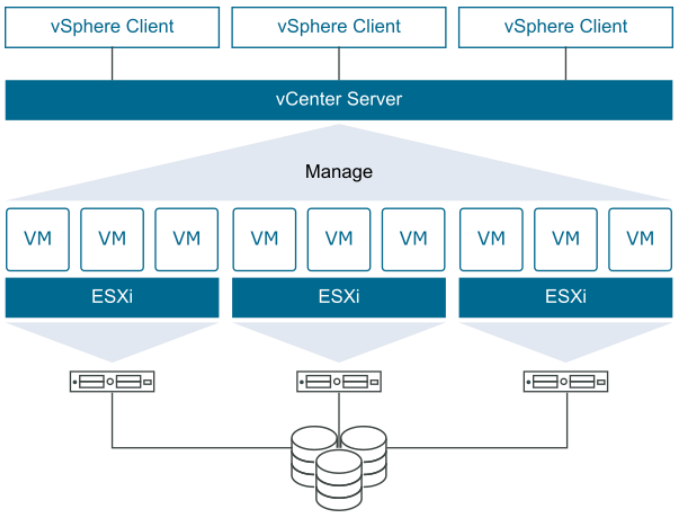
[13] IOPS = Input/Output Operations per Second
[14] Izvor: https://blogs.vmware.com/vsphere/2011/09/how-often-does-esxi-write-to-the-boot-disk.html

koristimo za izradu i upravljanje virtualnom okolinom. Moderna web aplikacija omogućava snalaženje i početnicima u virtualizaciji, a opcije često znaju imati i balončić koji opisuje rad i svrhu neke funkcije.

S obzirom na veliku funkcionalnost, visoku stabilnost te modernu web aplikaciju, ne čudi zašto se većina kompanija odluči upravo za VMware platformu kao rješenje za lokalnu virtualizaciju.

3.2. VMware vCenter Server

Dvije glavne komponente VMware vSphere virtualizacije su ESXi i vCenter Server. ESXi predstavlja virtualizacijsku platformu na kojoj se kreiraju i pokreću virtualne mašine, dok vCenter Server služi kao servis za upravljanje ESXi-em. Možemo ga zamisliti kao centralnog administratora koji definira koji od ESXi servera će odraditi specifičan zadatak, odnosno pokrenuti virtualnu mašinu. Web komponenta vCenter Server-a zove se vCenter Server Appliance, te preko nje pristupamo i upravljamo virtualnom okolinom. vCenter Server možemo instalirati na zasebnom serveru ili kao virtualnu mašinu unutar ESXi-a.



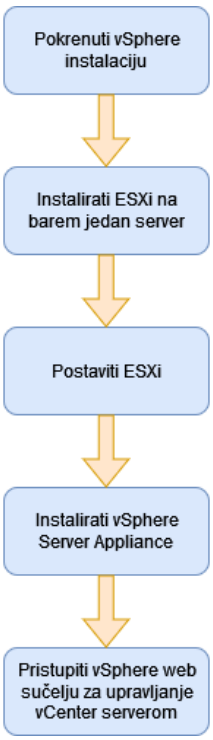
Slika 3.1 VMware vSphere komponente[15]

[15] Izvor: https://docs.vmware.com/en/VMware-vSphere

vCenter Server se instalira koristeći predkonfiguriranu virtualnu mašinu koja je optimizirana za pokretanje VMware vCenter Server-a. Tu virtualnu mašinu nije moguće editirati izuzev dodavanja procesora, radne memorije i diskovnog prostora. Komponente koje se nalaze u toj virtualnoj mašini su:

- Photon operacijski sustav
- PostgreSQL baza podataka
- vCenter Server platforma i njene komponente
- Ostali servisi koji su neophodni za licenciranje, certificiranje i jedinstvenu prijavu u sustav

Bez ESXi virtualizacijske platforme, vCenter Server nema realnu svrhu. Iz tog razloga se instalacija vrši tek nakon što smo postavili barem jedan ESXi server, kako je i prikazano sljedećim dijagramom:

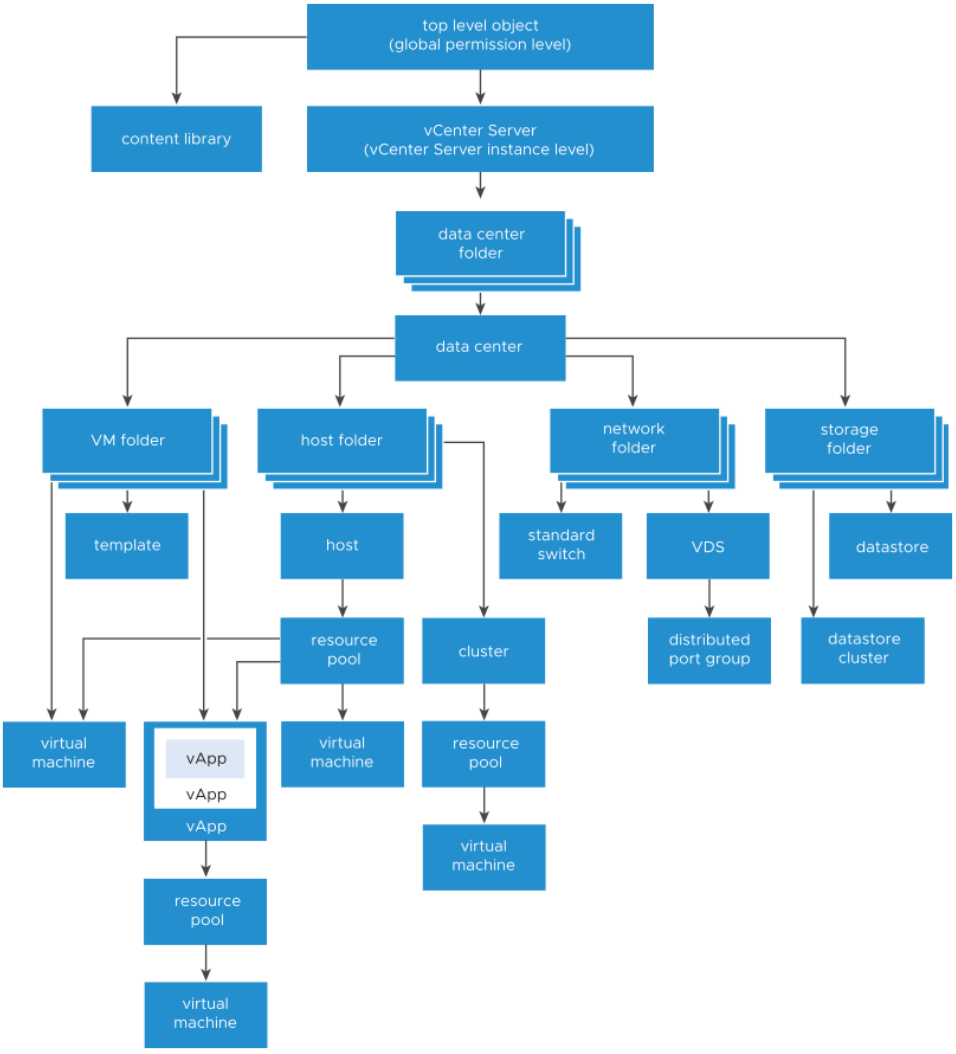


Slika 3.2 Redoslijed instalacije i konfiguracije za vSphere platformu

Sličan princip ćemo koristiti za podizanje naše testne okoline, izuzev vSphere instalacije koja će se nalaziti na zasebnom virtualnom serveru (neće se instalirati na ESXi server).

3.3. Hijerarhijska struktura u VMware virtualizaciji

Kreiranje virtualnih mašina i dodjeljivanje prava zahtjeva poznavanje hijerarhijske strukture objekata. Objekti unutar VMware vSphere-a su i korisnici i virtualne mašine, ali i mreža i direktorij, te nad njima primjenjujemo skup pravila koja definiraju razine pristupa do tog objekta. Shematski prikaz nasljeđivanja prava prikazan je sljedećim dijagramom:



Slika 3.3 Hijerarhijska struktura objekata unutar vSphere-a[16]

Odavde možemo vidjeti da se cijeli inventar sastoji od kolekcije podatkovnih centara (*eng. data center folder*). Unutar svakog podatkovnog centra imamo određene resurse na

[16] Izvor: https://docs.vmware.com/en/VMware-vSphere/

raspolaganju koji se prikazuju u obliku ukupne računarske snage, dostupne radne memorije te diskovnog prostora. Virtualne aplikacije i virtualne mašine se kreiraju unutar tih podatkovnih centara te ih možemo preseliti u direktorije kako bi ih grupirali i postigli odgovarajuću hijerarhijsku strukturu s kojom potom možemo upravljati. Slično kao i pri dijeljenju datoteka, umjesto da podijelimo jednu po jednu, najčešće podijelimo cijeli direktorij. Isti princip se ovdje koristi da bi se delegirala odgovornost i omogućio pristup nad više objekata za jednog ili grupu korisnika, umjesto da se za svaki objekt prava definiraju pojedinačno. Razumijevanje ovog pristupa biti će ključno za upravljanje tim podacima putem Python aplikacije.

Za početak ćemo objasniti vrste objekata te kako ćemo ih translatirati na našu okolinu.

- *Top-level object*: predstavljaju sve objekte kojima se upravlja putem vSphere platforme
- *Content library*: predstavlja kontejnerske objekte za virtualne mašine, aplikacije, tekstualne datoteke, ISO datoteke i sl. Koristi se za dijeljenje sadržaja između više vCenter instanci
- *vCenter Server*: predstavlja instancu vCenter Server-a.
- *Data Center folder*: predstavlja direktorij u kojem se nalazi grupa podatkovnih centara, npr. možemo kreirati direktorij naziva Zagreb i u njemu imati dva podatkovna centra naziva DC-Sesvete i DC-ZagrebZapad
- *Data Center*: predstavlja skup objekata koji se nalaze unutar tog podatkovnog centra. U njega se dodaju ESXi serveri, kreiraju virtualne mašine, mreže i diskovi.
- *VM Folder*: Predstavlja direktorij u koji možemo smjestiti više virtualnih mašina, od kojih neke mogu biti u uporabi, a neke služiti kao predložak za kreiranje novih
- *Virtual Machine*: Predstavlja pojedinačno virtualnu mašinu
- *vApp*: odnosi se na virtualne aplikacije koje se mogu nalaziti na virtualnim mašinama. Kao primjer možemo zamisliti internetski preglednik Google Chrome kojem se pristupa bez potrebe da se ulazi u virtualnu mašinu.
- *Resource Pool*: predstavlja logičku grupu resursa koja je dodijeljena skupu virtualnih mašina i aplikacijama na njima. Može se koristiti kada se želi ograničiti količina raspoloživih resursa za skup virtualnih mašina. Npr. kada ne želimo da aplikacije koje koristi web poslužitelja, a nalaze se na više virtualnih mašina, koriste ukupno više od 8GB radne memorije.
- *Host folder*: Kao i u slučaju kod direktorija virtualnih mašina, ovdje možemo grupirati ESXi servere koje koristimo za smještaj naših virtualnih mašina. Za primjer možemo

zamisliti organizaciju koja ima nekoliko katova na kojima se nalaze serveri te se onda oni smještaju u direktorije naziva „Prvi kat“, „Drugi kat“ itd.

- *Host*: Predstavlja instancu ESXi-a.
- *Cluster*: Vrsta objekta koja označava klaster ESXi servera. Koristi se kada se žele kreirati visoko dostupne virtualne mašine, dijeljena pohrana, mreže i sl.
- *Network Folder*: Služi za grupiranje virtualnih mrežnih preklopnika u jednom direktoriju.
- *Standard Switch*: Predstavlja objekt standardnog preklopnika koji se koristi za promet između virtualnih mašina. Potrebno ga je definirati na svakom ESXi serveru pojedinačno (kreira se prilikom instalacije).
- *VDS*: Predstavlja distribuirani preklopnik koji se koristi u klasteriranoj okolini. Neophodan je za kreiranje privatni virtualnih LAN okruženja i korištenje vMotion funkcije za nadzor mreže. Svi ESXi serveru imaju pristup na VDS dokle god su dio klastera.
- *Distributed Port Group*: Koristi se za konfiguraciju ulazno-izlaznog komunikacijskog sučelja na virtualnom mrežnom preklopniku. Unutar distribuirane grupe sučelja moguće je postaviti napredne mehanizme zaštite poput MAC provjere, vezanja uređaja za specifično sučelje, isključivanja sučelja i maksimalne brzine mrežnog prometa.
- *Storage folder*: Direktorij za grupiranje prostora za pohranu
- *Datastore*: Označava logički kontejner koji služi kao prostor za pohranu virtualnih mašina te ostale vrste podataka.
- *Datastore cluster*: Odnosi se na kolekciju logičkih kontejnera u kojoj se resursi dijele. Nad klasterom je moguće aktivirati „Storage DRS“ koji brine o tome da su virtualne mašine smještene na najbolje odgovarajuće diskove kako bi se maksimizirala efikasnost.

Korisnici sa administratorskim pravima definiranim na razini vCenter servera mogu upravljati svim objektima i pravima nad njima, izuzev *Content Library* dijela s obzirom da taj objekt ne spada pod vCenter Server. Za njega se treba dodijeliti posebno „Read-Only“ pravo na globalnoj razini.

4. Microsoft Windows Server konfiguracija

Za potrebe izrade virtualne okoline, a naposlijetku i same aplikacije, potrebno je dodati Windows Active Directory imenički sustav koji ćemo pokretati na Windows Server poslužitelju. S obzirom da je ovaj rad usmjeren na VMware virtualizaciju, nećemo pretjerano ulaziti u detalje Windows Server konfiguracije, već ćemo opisati strukturu imeničkog sustava odakle ćemo vući bazu korisnika koje ćemo koristiti na VMware vSphere platformi. Dodatno, upotrijebiti ćemo standardnu Windows 10 mašinu koju ćemo dodati u domenu kao bi verificirali mogućnost prijave u imenički sustav te na virtualizacijsku platformu.

Naš Windows poslužitelj sastojat će se od Microsoft Windows Server 2019 operacijskog sustava, sa 2 procesorske jezgre te 4 GB radne memorije. Za pohranu ćemo iskoristiti SSD sa 100GB slobodnog prostora, a IP adresa po izboru biti će 10.0.2.254.

4.1. Windows Active Directory imenički sustav

Imenički sustav je hijerarhijska struktura koja nam služi za pohranu informacija o objektima na mreži. Najlakše ju je poistovjetiti sa datotekama koje se nalaze unutar jedne mape, koja se pak nalazi unutar druge mape, koja se nalazi na C: disku. U ovom slučaju te informacije se tiču naziva korisnika koji se prijavljuje u sustav, lozinke koju koristi za prijavu, grupa kojima pripada unutar organizacije, kompjuteru na koji se prijavljuje itd. S obzirom da je u pitanju hijerarhijska struktura, do informacija je lako doći ako se zna putanja, što administratorima olakšava upravljanje okruženjem u kojem se ti korisnici nalaze.

Mi ćemo tu strukturu iskoristiti kako bi prvo kreirali, a potom povukli listu korisnika koje ćemo koristiti za prijavu u virtualnu okruženje, te kojima ćemo potom dodijeliti određene virtualne mašine za uporabu.

Radi lakšeg snalaženja, kreirati ćemo korisnike kojima ćemo ime postaviti na „Student“, a prezime na redni broj kreiranog korisnika. Tako da će naši korisnici biti „Student 1“, „Student 2“, „Student 3“, … Za metodu prijave koristiti ćemo oblik „imeprezime@vsphere.local“ što se za prvog korisnika translatira u „Student1@vsphere.local“. Lozinku ćemo svima postaviti na „Pa$$w0rd“ kako bi testirali mogućnost prijave u domensko okruženje.

Korisnike ćemo također podijeliti po grupama, a s obzirom da se naši korisnici zovu „Student“, prikladno ćemo ih svrstati po godinama. Prvih deset korisnika pripadati će grupi po imenu „1“, drugih deset po imenu „2“, trećih deset po imenu „3“, te tako sve do 50 korisnika. Dodatno, kreirati ćemo grupu Studenti koju ćemo pridružiti svim korisnicima. Na taj način imamo po deset godina na svakoj godini studija.

Za pregledniji prikaz, kreirati ćemo zasebne dvije organizacijske jedinice u koje ćemo smjestiti novokreirane objekte. Jedna će biti „Godina“, a druga „Studenti“. Kao što i sam naziv kaže, studente ćemo smjestiti u OU[17] „Studenti“, a godine u OU „Godina“

Kako bi malo ubrzali cijeli proces, djelomično ćemo automatizirati kreiranje korisnika. Organizacijske jedinice i grupe ćemo ručno kreirati, dok ćemo za korisnike upotrijebiti sljedeću PowerShell skriptu:

```
$Domain = 'vsphere.local'
$PW = 'Pa$$w0rd' | ConvertTo-SecureString -AsPlainText -Force

$GroupALL = Get-ADGroup -Identity "CN=Studenti,
OU=Godina,DC=vsphere,DC=local"
$Group1 = Get-ADGroup -Identity "CN=1,
OU=Godina,DC=vsphere,DC=local"
$Group2 = Get-ADGroup -Identity "CN=2,
OU=Godina,DC=vsphere,DC=local"
$Group3 = Get-ADGroup -Identity "CN=3,
OU=Godina,DC=vsphere,DC=local"
$Group4 = Get-ADGroup -Identity "CN=4,
OU=Godina,DC=vsphere,DC=local"
$Group5 = Get-ADGroup -Identity "CN=5,
OU=Godina,DC=vsphere,DC=local"


foreach($i in 1..10)
{
    $Name = 'Student{0}' -f $i
```

[17] OU = Organizacijska jedinica (Organizational Unit)

```
    New-ADUser -Name $Name -AccountPassword $PW -GivenName
"Student" -Surname $i -UserPrincipalName ($Name + '@' +
$Domain) -Path "OU=Studenti,DC=vsphere,DC=local" -Department
"Studenti" -Enabled:$true
    Add-ADGroupMember -Identity $Group1 -Members $Name
    Add-ADGroupMember -Identity $GroupALL -Members $Name

}
foreach($i in 11..20)
{
    $Name = 'Student{0}' -f $i

    New-ADUser -Name $Name -AccountPassword $PW -GivenName
"Student" -Surname $i -UserPrincipalName ($Name + '@' +
$Domain) -Path "OU=Studenti,DC=vsphere,DC=local" -Department
"Studenti" -Enabled:$true
    Add-ADGroupMember -Identity $Group2 -Members $Name
    Add-ADGroupMember -Identity $GroupALL -Members $Name
}
foreach($i in 21..30)
{
    $Name = 'Student{0}' -f $i

    New-ADUser -Name $Name -AccountPassword $PW -GivenName
"Student" -Surname $i -UserPrincipalName ($Name + '@' +
$Domain) -Path "OU=Studenti,DC=vsphere,DC=local" -Department
"Studenti" -Enabled:$true
    Add-ADGroupMember -Identity $Group3 -Members $Name
    Add-ADGroupMember -Identity $GroupALL -Members $Name
}
foreach($i in 31..40)
{
    $Name = 'Student{0}' -f $i

    New-ADUser -Name $Name -AccountPassword $PW -GivenName
"Student" -Surname $i -UserPrincipalName ($Name + '@' +
$Domain) -Path "OU=Studenti,DC=vsphere,DC=local" -Department
"Studenti" -Enabled:$true
    Add-ADGroupMember -Identity $Group4 -Members $Name
    Add-ADGroupMember -Identity $GroupALL -Members $Name
}
```

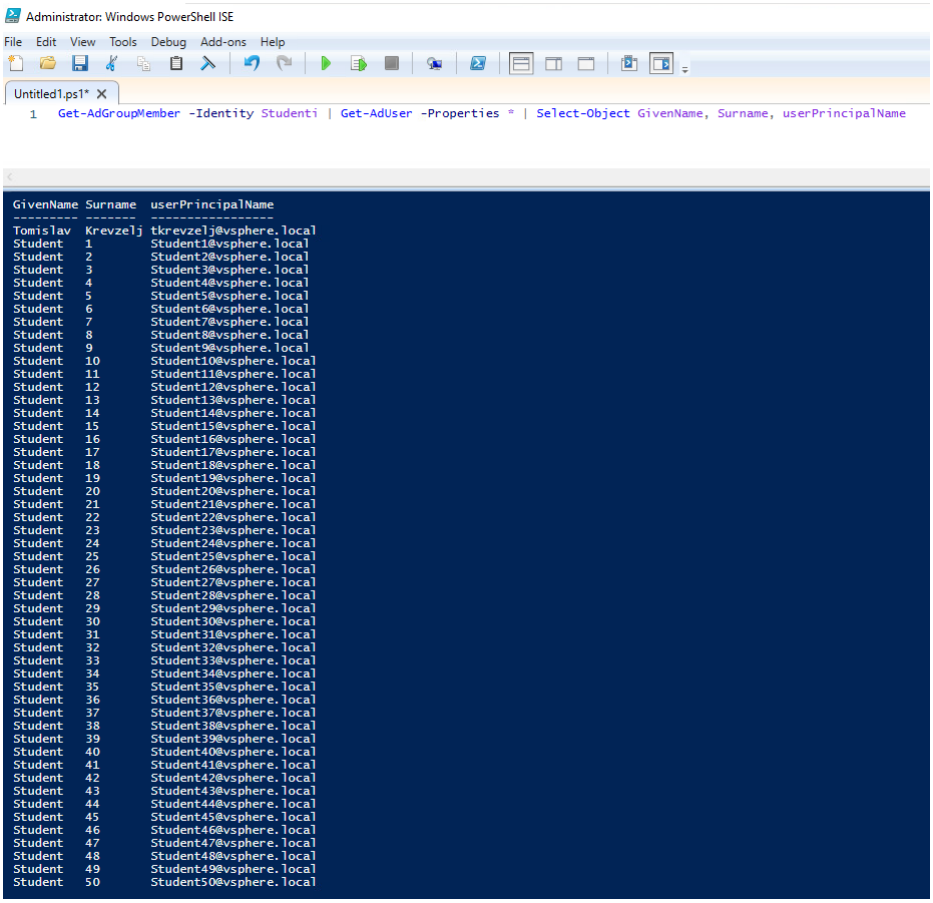
```
foreach($i in 41..50)
{
    $Name = 'Student{0}' -f $i

    New-ADUser -Name $Name -AccountPassword $PW -GivenName
"Student" -Surname $i -UserPrincipalName ($Name + '@' +
$Domain) -Path "OU=Studenti,DC=vsphere,DC=local" -Department
"Studenti" -Enabled:$true
    Add-ADGroupMember -Identity $Group5 -Members $Name
    Add-ADGroupMember -Identity $GroupALL -Members $Name
}
```

Kôd 4.1 PowerShell skripta za kreiranje korisnika u imeničkom sustavu

Uspješnost kreiranja korisnika možemo provjeriti na više načina, gdje bi jedan od njih bio korištenje alata unutar Windows Server operacijskog sustava pod nazivom „Active Directory Users and Computers“. Tamo možemo ručno otvoriti odgovarajuće organizacijske jedinice te grupe i korisnike pojedinačno. Druga opcija bi bila upotrijebiti PowerShell kod koji će uz odgovarajući upit izlistati korisnike koji su prisutni u imeničkom sustavu. Ostale opcije bi uključivale korištenje trećih alata koji se spajaju na poslužitelj, no ovdje ih nećemo imenovati te ćemo radi preglednijeg prikaza ovdje prikazati rezultat PowerShell upita za naredbu:

```
Get-AdGroupMember -Identity Studenti | Get-AdUser -Properties * | Select-
Object GivenName, Surname, userPrincipalName
```

Naredbu je moguće izvršiti unutar regularnog PowerShell okvira ili korištenjem PowerShell ISE aplikacije koja će potencijalne greške u sintaksi automatski podcrtati te ponuditi adekvatno rješenje. Rezultat gore navedenog upita prikazan je sljedećom slikom unutar PowerShell ISE aplikacije.



Slika 4.1 Korisnici kreirani PowerShell skriptom

Po pridruživanju računala u domenu, korisnici se mogu prijaviti korištenjem bilo kojeg „userPrincipalName“ za pristup računalu i domenskim resursima.

4.2. DNS sustav

Prije samog podizanja okoline, potrebno je postaviti DNS sustav koji ćemo koristiti za pridruživanje imena poslužiteljima. S obzirom da instalacija vSphere komponente uz adresni dio zahtjeva i imenski dio, moramo postaviti zapise koji će povezati IP adresu i naziv samog računala. U tu svrhu koristiti ćemo DNS servis unutar Microsoft Server 2019 operacijskog sustava na kojem se nalazi i imenički sustav.

DNS, slično kao i kod imeničkog sustava, je distribuirani hijerarhijski sustav gdje se nalaze informacije o povezanosti između IP adresa i njihovih simboličkih imena. Prilikom pretraživanja web adresa poput www.google.com, naše računalo šalje upit DNS poslužitelju

za adresu te domene. DNS ju potom vraća u obliku IP adrese poput 142.251.39.46, te se daljnja komunikacija odvija direktno sa Google poslužiteljem.

U našem slučaju, s obzirom da je riječ o sofisticiranom softveru, DNS nam treba za kreiranje i verifikaciju certifikata prilikom instalacije virtualne okoline. Kako je današnja komunikacija kriptirana, upotrebom certifikata možemo verificirati da je osoba ili poslužitelj s druge strane uistinu ona za koju se izdaje. Prilikom instalacije vSphere komponente, možemo skinuti i spremiti taj certifikat kako bi prilikom pristupa bili sigurni da pristupamo na ispravnu platformu.

S obzirom da je DNS sustav autoritativan za vsphere.local, unutar njega ćemo kreirati šest „A“ zapisa. „A“ zapis povezuje IP adresu sa nazivom i domenom računala. Iako je moguće imati više zapisa koji pokazuju na istu IP adresu, u našem slučaju će svaki pokazivati na svoju adresu. Za adresni raspon koristi ćemo subnet 10.0.2.0/24, a adrese ćemo dodijeliti na sljedeći način:

Tablica 4.1 DNS zapisi unutar vsphere.local domene

Naziv	FQDN[18]	Adresa
cli	cli.vsphere.local	10.0.2.100
dc	dc.vsphere.local	10.0.2.254
esxi1	esxi1.vsphere.local	10.0.2.11
esxi2	esxi2.vsphere.local	10.0.2.12
linux	linux.vsphere.local	10.0.2.200
vcsa	vcsa.vsphere.local	10.0.2.10

Detalje oko rasporeda adresa ćemo pokriti u sljedećem poglavlju gdje se bavimo pripremom okoline za virtualizacijsku platformu.

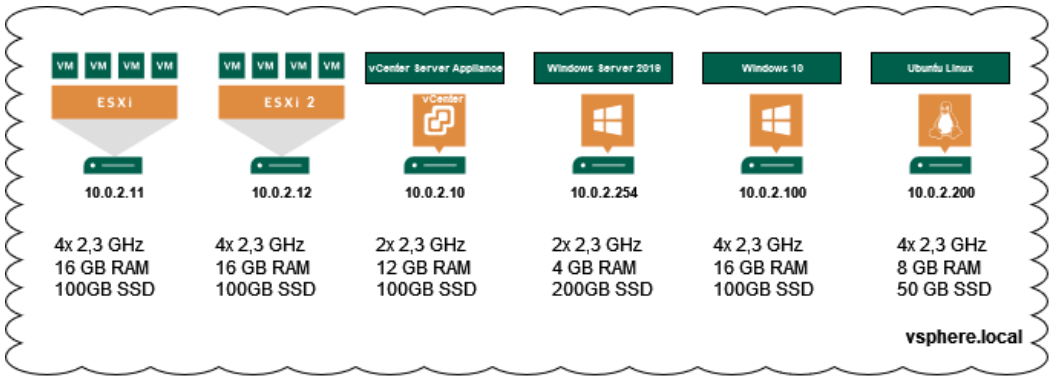
[18] FQDN = Fully Qualified Domain Name

5. Priprema virtualne okoline

Za podizanje virtualne okoline upotrijebiti ćemo 3 servera, od kojih će 2 služiti kao ESXi serveri, a jedan će biti za instalaciju vCenter servera. Svaki od servera imati će po 4 procesora takta 2.3 GHz te 16 GB radne memorije. Uređaj za pohranu će biti SSD[19] tipa, sa 100 GB prostora za smještaj virtualnih mašina, te će svaki od njih imati po jednu mrežnu karticu s pripadajućom IP adresom. S obzirom na nedostatak hardverskih resursa, okolina će biti podignuta u ugniježdenoj virtualizaciji (*eng. nested virtualization*) gdje se virtualne mašine nalaze unutar virtualne mašine. Iako se taj oblik virtualizacije u praksi ne koristi toliko često, nama je potreban za izradu virtualne okoline jer bi u protivnom morali imati fizičke uređaje kako bi simulirali realno okruženje neke organizacije. Domena koju koristimo je vsphere.local, a mrežne adresa po izboru su iz 10.0.2.0/24 subneta.

Uz VMware dio za virtualizaciju, imamo Windows Server 2019 poslužitelj na kojem se nalaze domena i DNS sustav, klijentsko računalo sa Windows 10 operacijskim sustavom za prijavu u domenu te Linux računalo sa Ubuntu operacijskim sustavom za izradu aplikacije.

IP adrese i DNS nazive postavljamo temeljem već definirane tablice, a shematski prikaz, ako zanemarimo dodatni sloj virtualizacije, bio bi sljedeći:



Slika 5.1 Virtualna okolina za smještaj virtualnih mašina

Gore navedene resurse moguće je prilagoditi vlastitom scenariju, pritom imajući u vidu minimalne hardverske zahtjeve za operacijske sustave koji će se u tom okruženju pokretati.

[19] SSD = Solid state drive

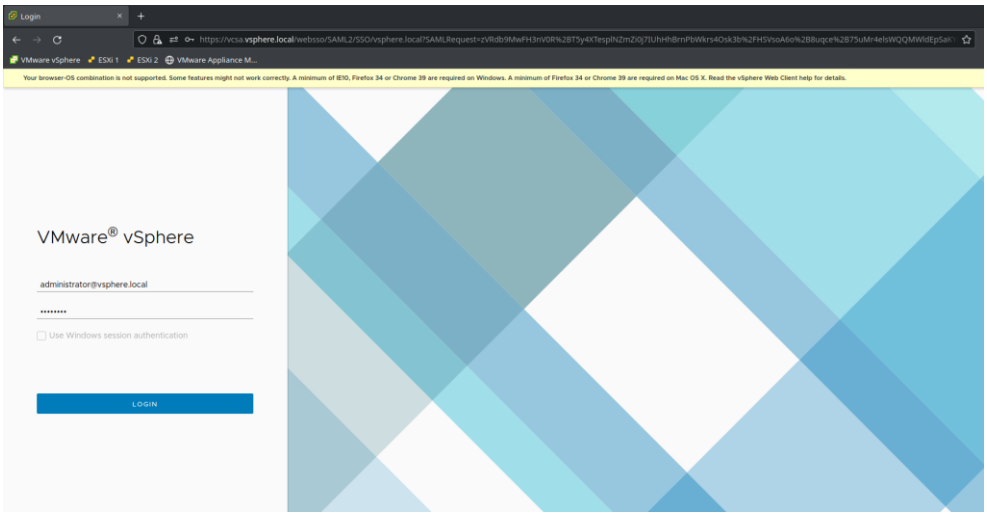
6. Izrada i konfiguracija virtualnih mašina

Virtualne mašine su glavne komponente virtualne infrastrukture. Pri izradi, asociramo ih s specifičnim logičkim kontejnerom za pohranu, operacijskim sustavom te virtualnim hardverom kojim će se ta virtualna mašina koristiti. Sav virtualni hardver pruža identične funkcije kao i fizički, a virtualna mašina ima pristup procesoru, memoriji, pohrani i mreži koja se nalazi na serveru na kojem je i pokrenuta.

U ovom poglavlju proći ćemo kroz izradu virtualne mašine putem web grafičkog sučelja, dok ćemo u idućem istu funkcionalnost postići putem Python aplikacije.

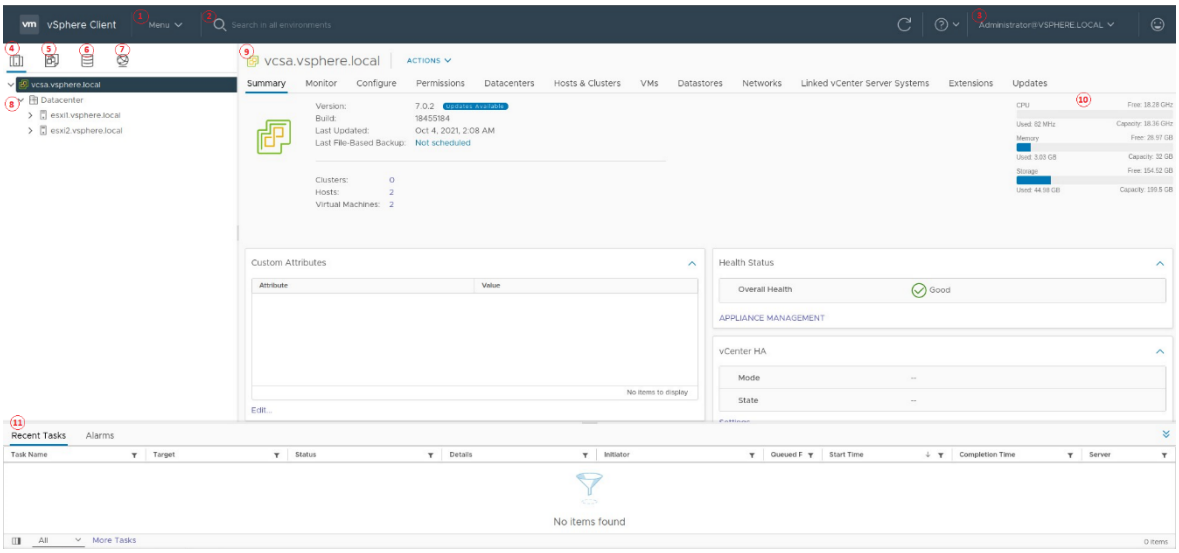
Ukoliko je okolina postavljena kao što je navedeno u prethodnom poglavlju, web sučelju aplikacije možemo pristupiti putem web adrese: https://vcsa.vsphere.local/ui

Prijavu vršimo korištenjem administratorskog korisnika kojeg smo kreirali prilikom instalacije vSphere-a, što je administrator@vsphere.local te za lozinku upisujemo Pa$$w0rd.



Slika 6.1 Prijava u vSphere sučelje

Nakon uspješne autentifikacije, dočekuje nas prozor sa dostupnim resursima unutar naše virtualne okoline. U njemu možemo iščitati osnovne informacije o našoj virtualnoj okolini kao što su verziji softvera koju koristimo, količini servera i virtualnih mašina, broju procesora, količina radne memorije i pohrane. Također možemo iščitati trenutne zadatke koji se u ovom trenu izvršavaju unutar naše virtualne okoline.



Slika 6.2 Dostupne opcije unutar vSphere sučelja

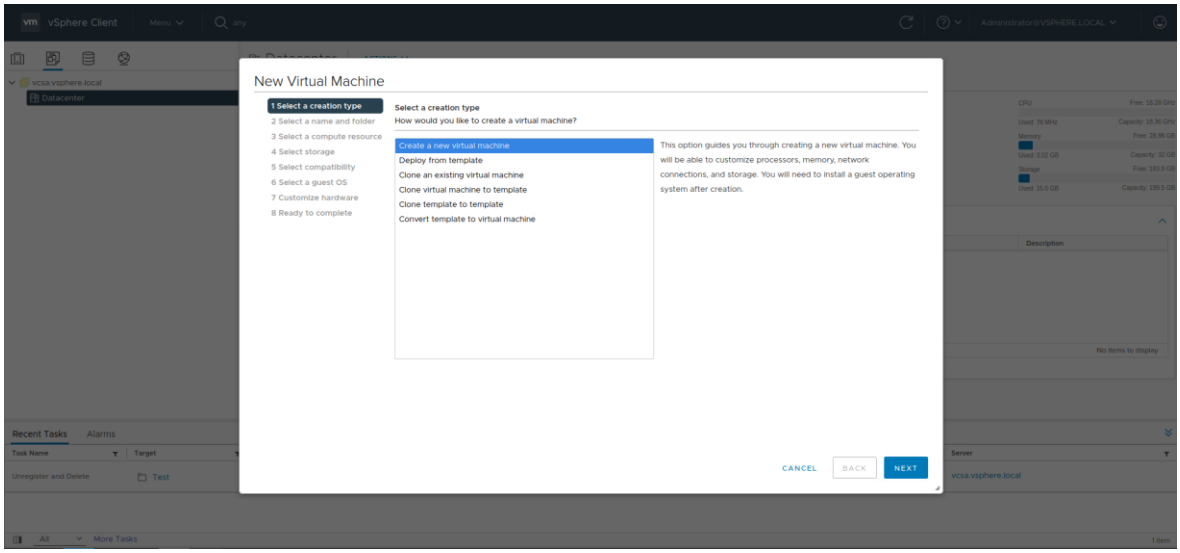
1. Unutar glavnog menija možemo se prebacivati između pogleda na servere, virtualne mašine, pohranu i mreže. Također možemo pristupiti biblioteci sadržaja, definiranju određenih politika, ali i samoj administraciji servera
2. Unutar tražilice možemo pretraživati sve objekte unutar virtualne okoline poput virtualnih mašina ili mreža, ali isto tako i određenih karakteristika poput Windows ili Linux operacijskog sustava, ili pak svih objekata koji odgovaraju određenom kriteriju.
3. Unutar korisničkog menija možemo promijeniti lozinku, temu, željene postavke ili se odjaviti iz sustava
4. Meni koji nam prikazuje dostupne ESXi servere unutar našeg virtualnog okruženja
5. Meni za prikaz virtualnih mašina
6. Meni za prikaz pohrane
7. Meni za prikaz virtualnih mreža
8. Popis dostupnih ESXi servera unutar našeg podatkovnog centra
9. Informacije o verziji i zdravlju vSphere komponente, količini ESXi servera, klastera i virtualnih mašina
10. Ukupno dostupna i iskorištena količina resursa unutar odabranog podatkovnog centra
11. Popis nedavno izvršenih zadataka (kreiranje resursa, promjena konfiguracije, nadogradnja sustava itd.) i onih koji se trenutno izvršavaju

Prije same izrade virtualne mašine, moramo se uvjeriti da imamo dovoljno resursa koje ta virtualna mašina zahtjeva. To u osnovi znači dovoljno procesorske snage, radne memorije te diskovnog prostora za smještaj virtualne mašine. Također, na virtualne mašine želimo smjestiti određene aplikacije kojima želimo pristupiti izvana ili instalirati aplikacije za praćenje stabilnosti rada drugih aplikacija te nam iz tog razloga treba i mrežna komponenta.

Primjerice za Windows Server 2019 potreban je 64-bitni procesor brzine 1.4 GHz, minimalno 512 MB radne memorije te 32 GB diskovnog prostora[20]. Pogledom na našu infrastrukturu vidimo da te uvjete ispunjavamo.

Za instalaciju virtualne mašine potrebna nam je još i ISO datoteka koja će se nalaziti na ESXi-u ili OVF predložak kojeg se može iskoristiti za dodavanje već kreiranih virtualnih mašina. Za prvu instalaciju koristiti ćemo Windows Server 2019 ISO datoteku smještenu na ESXi – 1 serveru.

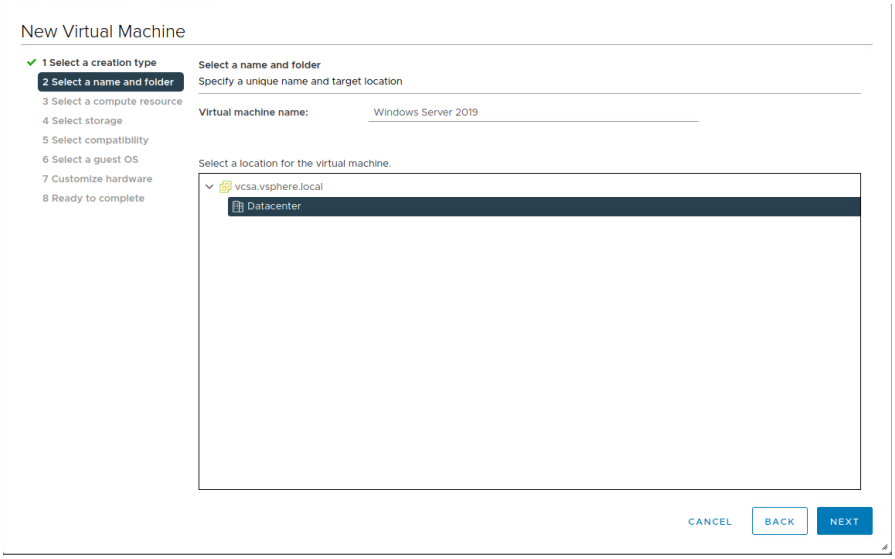
Prebacivanjem pogleda na virtualnih mašine te desnim klikom na „*Datacenter*“ možemo odabrati „*New Virtual Machine*“ te pokrenuti čarobnjak za izradu virtualne mašine.



Slika 6.3 Čarobnjak za izradu virtualne mašine

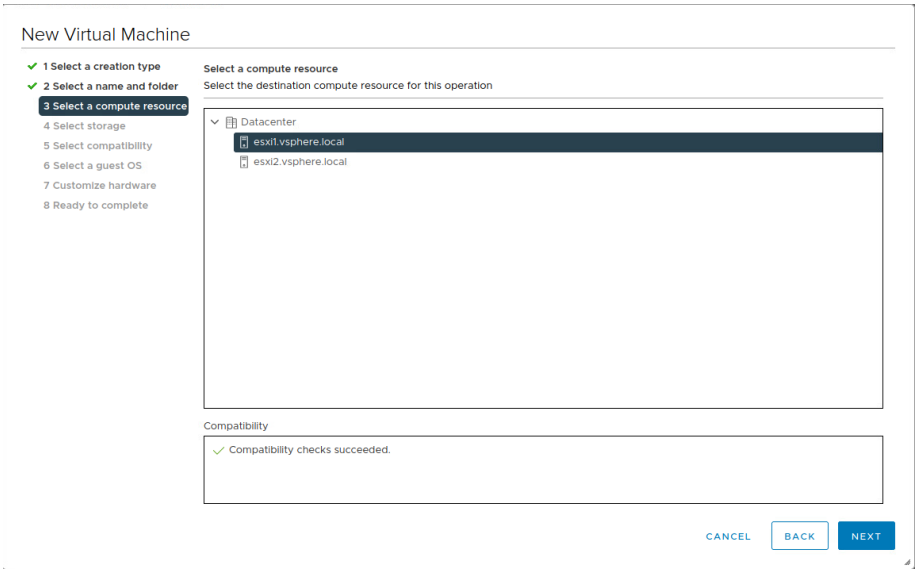
[20] Izvor: https://learn.microsoft.com/en-us/windows-server/get-started/hardware-requirements

Odavde biramo izradu nove virtualne mašine klikom na „*New Virtual Machine*“ kako bismo mogli instalirati našu virtualnu mašinu korištenjem ISO datoteke. Za naziv ćemo odabrati Windows Server 2019, a lokacija VM će biti dostupni podatkovni centar.



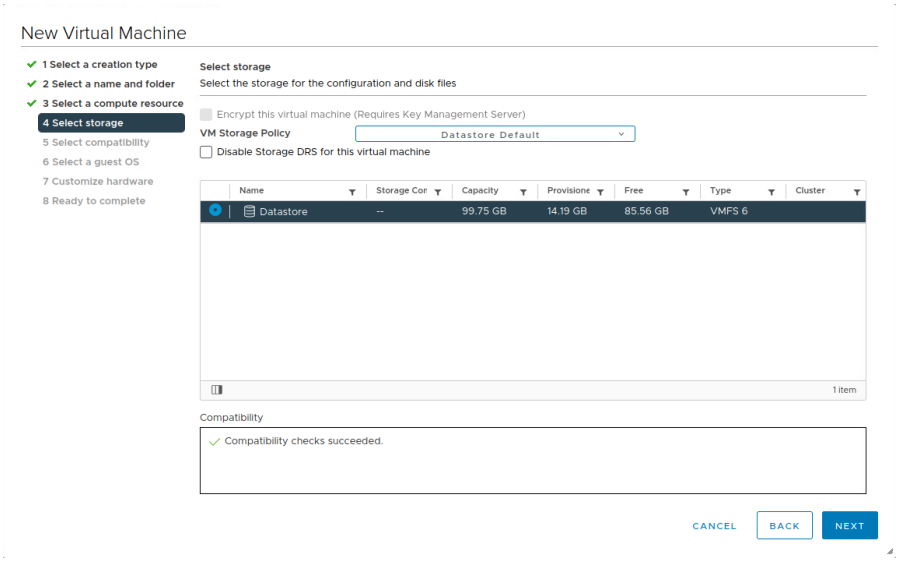
Slika 6.4 Odabir naziva virtualne mašine

Server na kojem će se virtualna mašina pokretati biti će esxi1.vsphere.local.



Slika 6.5 Odabir ESXi servera na kojem će se virtualna mašina pokretati

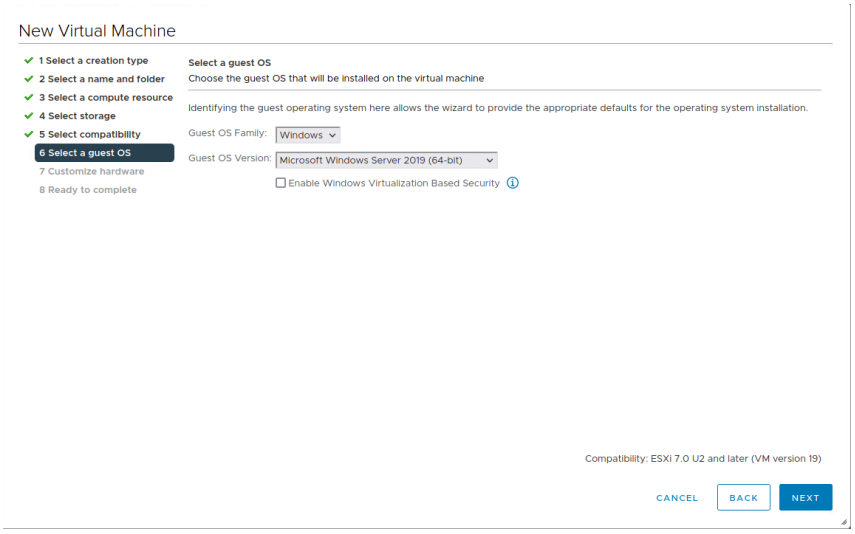
Za smještaj podataka od virtualne mašine koristiti ćemo pohranu dostupnu na ESXi – 1 serveru. Odavde možemo primijetiti da s obzirom da se ne koristi udaljeni prostor za pohranu, odabirom ESXi – 1 servera vidimo isključivo diskovni prostor koji se nalazi na ESXi – 1 serveru (*Datastore*). Da smo odabrali ESXi – 2 vidjeli bi isključivo prostor koji se nalazi na drugom serveru – Datastore2. S obzirom da nije riječ o produkciji, koristiti ćemo predefiniranu politiku postavki za pohranu. Kreiranjem ili modificiranjem politika možemo utjecati na replikaciju i keširanje podataka, te na količinu I/O operacija na disku koje će biti dostupne za tu virtualnu mašinu.



Slika 6.6 Odabir postavki za pohranu virtualne mašine

Kompatibilnost je potrebno mijenjati u slučaju kada se ESXi i vSphere nalaze na različitim verzijama ili željena virtualna mašina ne podržava najnoviju inačicu VMware virtualizacije. Kao primjer možemo uzeti stare operacijske sustave i hardver za koji ne postoje novi upravljački programi. U praksi to predstavlja sigurnosni problem i potencijalnu nekompatibilnost određenih funkcija te je iz tog razloga potrebno nadograditi sustav. S obzirom dam u trenutku pisanja, koristimo najnovije inačice softvera, te postavke ne diramo.

U odabir operacijskog sustava biramo Windows familiju operacijskih sustava, a za verziju Microsoft Windows Server 2019 (64-bit).



Slika 6.7 Odabir operacijskog sustava

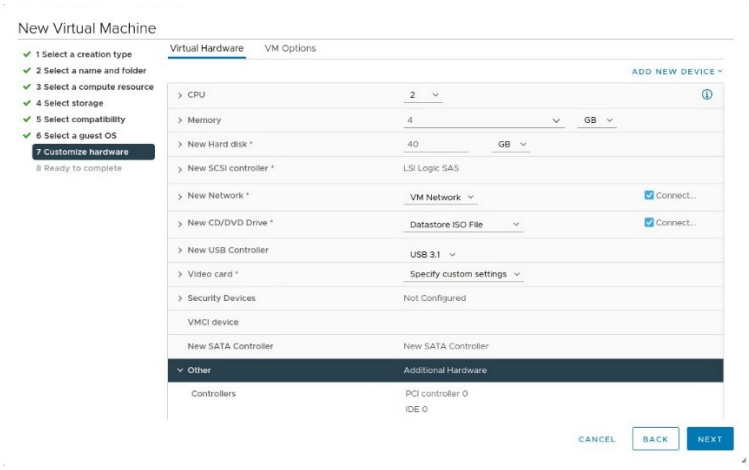
Sljedeći izbornik predstavlja najbitniji dio kod kreiranja virtualnih mašina, a to je odabir hardvera. Iako u praksi operacijski sustavi rade čak i ako im se dodijeli manje od minimalno definirane količine resursa, odabirom manje od preporučene količine rezultirati će usporavanjem sustava. Naravno, isto tako ne treba pretjerivati s količinom resursa jer u tom slučaju ostaju neiskorišteni i nedostupni drugim virtualnim mašinama. Iz tog razloga za Windows Server 2019 koristiti ćemo 2 jezgre, 4 GB radne memorije te 40 GB diskovnog prostora. Za mrežu ćemo odabrati VM Network, a CD/DVD će biti ISO datoteka operacijskog sustava.

Naprednim postavkama virtualne mašine možemo omogućiti dodavanje procesora i radne memorije u radu te ih rezervirati djelomično ili u potpunosti. Nad procesorom je moguće uključiti i omogućiti ugniježdenu virtualizaciju, ali i limitirati korištenje ciklusa kako bi dodijelili npr. 1.5 jezgru. Kod postavki tvrdog diska možemo definirati da li će se sav prostor zauzeti u potpunosti (*eng. Thick provision*) ili će se prezentirati operacijskom sustavu u jednoj veličini, dok će se u stvarnosti popunjavati ovisno o količini podataka na njemu (*eng. Thin provision*). Naravno, moguće je definirati količinu IOPS-a te lokaciju gdje će se taj disk nalaziti. Od dodatnih funkcija moguće je birati vrstu mrežnog adaptera i njegovu MAC[21] adresu, te postavke grafike u slučaju kada želimo proslijediti fizičku karticu virtualnoj mašini

[21] MAC = Media Access Control

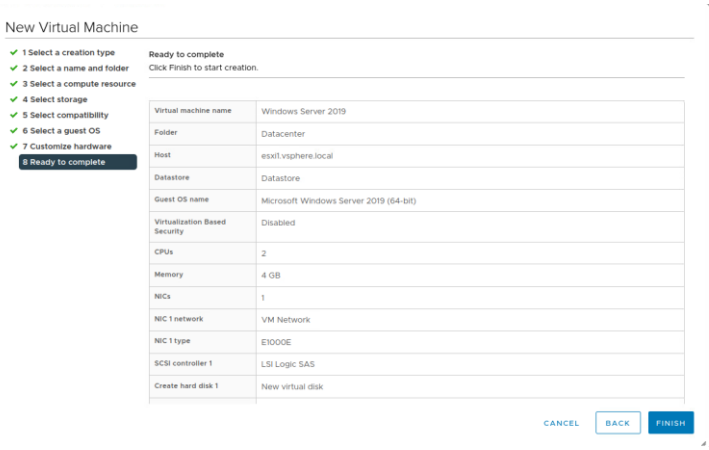
te dodijeliti određenu količinu memorije. Moguće je dodati i dodatne kontrolere, diskove i ostale uređaje te ih konfigurirati po želji.

Unutar postavki virtualne mašine može se definirati i kriptiranje datoteka, vrsta BIOS-a te funkcije poput *Wake-on-Lan* i mirovanja virtualne mašine nakon perioda neaktivnosti.



Slika 6.8 Postavke virtualnog hardvera

Po završetku, dolazimo do sumariziranog prikaza postavki te potvrđujemo kreiranje virtualne mašine.



Slika 6.9 Sumarizirani prikaz postavki virtualne mašine

Progres izrade pratimo putem *Recent Tasks* okvira, te čekamo da se status virtualne mašine promijeni na *Completed,* što označava uspješnu kreiranje iste.

6.1. Postavke za pripremu virtualne okoline

S obzirom da bi kompletan vodič kroz instalaciju oduzeo previše vremena i nepotrebno oduzeo prostor u radu, spomenuti ćemo osnovne postavke za kreiranje okruženja, a za detaljne upute potrebno je konzultirati se s dodatnim materijalom.

Okolinu ćemo konfigurirati na sljedeći način:

- dc.vsphere.local
  - Windows Server 2019 instalacija sa osnovnim postavkama
  - Kreiran korisnik „Administrator“ sa lozinkom „Pa$$w0rd“
  - Podignuta domena sa osnovnim postavkama naziva vsphere.local te lozinkom za oporavak „Pa$$w0rd“
  - Kreiran domenski administrator „VSPHERE\Administrator“ sa lozinkom „Pa$$w0rd“
  - Kreirane dvije organizacijske jedinice pod nazivom „Studenti“ i „Godina“
  - Kreirano 50 korisnika imena „Student“ i prezimena od 1 do 50
  - Kreirane grupe „Studenti“, „1“, „2“, „3“, „4“ te „5“, gdje se u brojčanim grupama nalazi po 10 korisnika, a u grupi „Studenti“ svi korisnici
  - DNS sustav sa zapisima za vsphere.local domenu za dc, esxi1, esxi2, vcsa, cli te linux računala
  - Postavljen domenski naziv „dc“ te statička IP adresa 10.0.2.254/24
- esxi1.vsphere.local i esxi2.vsphere.local
  - Instalacija verzije 7.0.3 putem ISO datoteke
  - Korisniku „root“ dodijeljena lozinka „Pa$$w0rd“
  - Postavljeni domenski nazivi esxi.vsphere.local te esxi2.vsphere.local te pripadajuće IP adrese 10.0.2.11/24 i 10.0.2.12/24
  - Kreiran „Datastore“ i „Datastore2“ za pohranu virtualnih mašina
- vcsa.vsphere.local
  - Instalacija verzije 7.0.3 putem vSphere Server Appliance ISO datoteke sa udaljenog računala.
  - Naziv virtualne mašine „VMware vCenter Server“ sa „root“ lozikom „Pa$$w0rd“
  - Instalacija oblika „tiny“ sa 2 jezgre te 12 GB radne memorije
  - Statička IP adresa 10.0.2.10/24 sa FQDN nazivom „vcsa.vsphere.local“ i DNS serverom 10.0.2.254 (dc.vsphere.local)
  - Kreiran korisnik „Administrator@vsphere.local“ za upravljanje vSphere platformom.
  - Kreiran podatkovni centar naziva „Datacenter“
  - Dodani ESXi serveri putem „root“ korisnika

- cli.vsphere.local
  - Windows 10 Pro instalacija sa osnovnim postavkama
  - Postavljen naziv „cli“ te IP adresa 10.0.2.100 i DNS server 10.0.2.254
  - Dodan u domensko okruženje korištenjem „VSPHERE\Administrator“ korisnika
- linux.vsphere.local
  - Linux Ubuntu 21.10 instalacija
  - Instalirani python3 te PyCharm za pisanje aplikacije
  - Instaliran GOVC i pyVmomi moduli za upravljanje vSphere-om
  - Instaliran python-ldap paket za pristup imeničkom sustavu na DC[22]-u.

Redoslijed instalacije vrši se na način da se prvo instalira i konfigurira server na kojem je domenski kontroler, potom ESXi 1 i 2 serveri, zatim *vSphere Server Appliance* i naposlijetku Windows 10 i Linux računalo.

Po završetku svih koraka, spremni smo za kreiranje virtualnih mašina na novostvorenoj virtualnoj okolini.

[22] DC = Domain Controller

6.2. Izrada predloška virtualne mašine

Virtualne mašine možemo pretvoriti u predloške kako bi od njih kreirali nove virtualne mašine. Time štedimo vrijeme i resurse jer se cijeli proces kreiranja virtualne mašine svodi na par klikova.

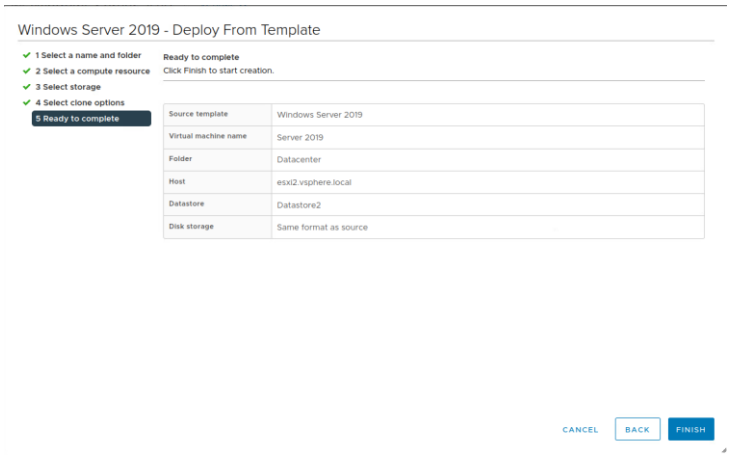
Unutar vSphere sučelja predložak virtualne mašine možemo kreirati na 3 načina:

- Pretvaranjem virtualne mašine u predložak
- Kloniranjem virtualne mašine u predložak
- Kloniranjem predloška

Pretvaranjem virtualne mašine u predložak onemogućujemo daljnje modificiranje te virtualne mašine. Predložak nije moguće upaliti niti mu promijeniti postavke sve dok se ne vrati u oblik virtualne mašine. Kloniranjem virtualne mašine u predložak stvaramo novu kopiju te virtualne mašine bez da utječemo na original. Na taj način možemo koristiti originalnu virtualnu mašinu dok predložak možemo koristiti za kreiranje novih virtualnih mašina. Kloniranjem predloška stvaramo novi predložak, odnosno kopiju originalnog predloška.

Prije kreiranja predloška Windows Server 2019, poželjno je pokrenuti Windows alat po imenu *Sysprep* kako bi se resetirale sigurnosne identifikacijske postavke virtualne mašine te kako bi ih mogli ubuduće razlikovati na mreži. Također, po završetku treba ugasiti virtualnu mašinu i tek onda od nje kreirati predložak na način da se desnim klikom na virtualnu mašinu odabere „*Template*“ te potom „*Convert to Template*“.

Potom tu virtualnu mašinu možemo koristiti za kreiranje novih virtualnih mašina čak i na druge servere na način da desnim klikom odaberem predložak i odaberemo „*New VM from This Template*“. U slučaju da ne želimo mijenjati hardverske postavke, sve što moramo odabrati je novi naziv virtualne mašine, server koji će ju pokretati te lokaciju pohrane datoteka.



Slika 6.10 Izrada virtualne mašine korištenjem predloška

Po završetku, virtualna mašina je spremna za uporabu. Prilikom prvog pokretanja potrebno je upisati novi ključ proizvoda te nove korisničke podatke za administratorski račun. Pri instalaciji će se generirati novi unikatni SID[23] broj koji će potom služiti za raspoznavanje te virtualne mašine od svih budućih koje se kreiraju putem istog predloška.

6.3. Izrada snimke virtualne mašine

Izradom snimke virtualne mašine bilježimo trenutno stanje i podatke unutar te virtualne mašine. Snimka je kompletna kopija virtualne mašine koja se potom može iskoristiti za migriranje na druge servere i pokretanje više instanci iste virtualne mašine ili za vraćanje u prvobitno stanje nakon određenih izmjena.

Najčešće se kreira kada se postigne operativno stanje virtualne mašine kao točka na koju se uvijek možemo vratiti u slučaju ako se nešto raspadne. Periodičkom izradom snimki tako možemo bilježiti više operativnih stanja te se uvijek vratiti na zadnje funkcionalno. Na taj način možemo testirati instaliranje aplikacija i paketa za koje nismo sigurni da li će utjecati na stabilnost sustava.

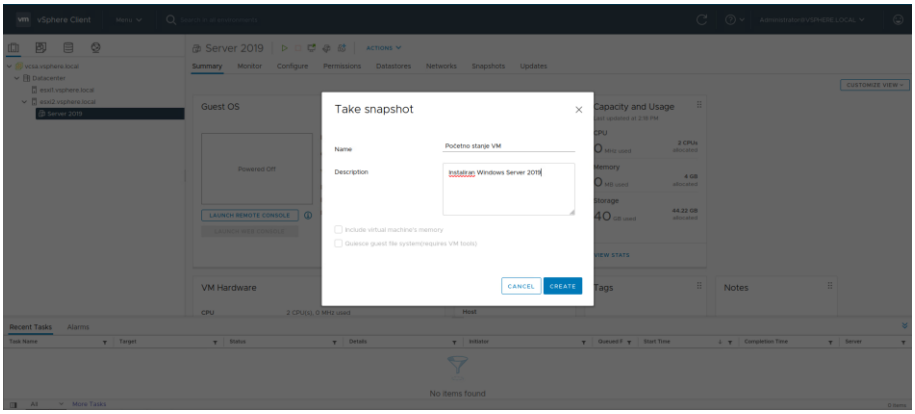
U trenutku okidanja snimke sustava spremamo sliku sustava te kreiramo novu datoteku u koju će se dalje spremati promjene koje izvršimo nad tom virtualnom mašinom. Te promjene mogu biti instalacija aplikacija, brisanje datoteka, ali čak i dodavanje ili uklanjanje određenog hardvera unutar postavki virtualne mašine.

[23] SID = Security Identifier

Moguće je kreirati više snimki sustava te se u tom slučaju svaka snimka nadovezuje na onu prethodnu. Na taj način kreiramo roditelj-dijete vezu u razgranatom obliku gdje se svaka promjena sprema na novi delta disk. Ti diskovi spremaju se u direktorij gdje se nalazi konfiguracija virtualne mašine.

Unutar snimke bilježe se hardverske postavke virtualne mašine, operativno stanje (uključeno, isključeno, u hibernaciji), podaci na disku te stanje memorije. Kreiranjem više snimki sustava možemo negativno utjecati na performanse sustava, te se iz tog razloga preporuča kreiranje 2 do maksimalno 3 snimke[24]. Kako bi vratili izvorne performanse virtualne mašine, snimke je moguće konsolidirati u jednu. Pritom se brišu redundantni diskovi čime se štedi na prostoru, a kreiranje sigurnosne kopije je lakše i brže putem odgovarajućih alata.

Snimku virtualne mašine možemo kreirati na način da desnim klikom odaberemo virtualnu mašinu te unutar „*Snapshots*“ okvira odaberemo „*Take Snapshot*“. Odande se otvara okvir gdje je moguće upisati naziv snimke te opis promjena.



Slika 6.11 Kreiranje snimke virtualne mašine

Izrađenim snimkama možemo pristupiti desnim klikom na virtualnu mašinu te unutar „*Snapshots*“ okvira odabirom „*Manage Snapshots*“. Isto tako možemo i samo kliknuti na virtualnu mašinu te unutar alatne trake navigirati na „*Snapshots*“ dio. Tamo možemo promijeniti naziv i opis snimki, vratiti virtualnu mašinu u početno stanje te obrisati kreirane snimke.

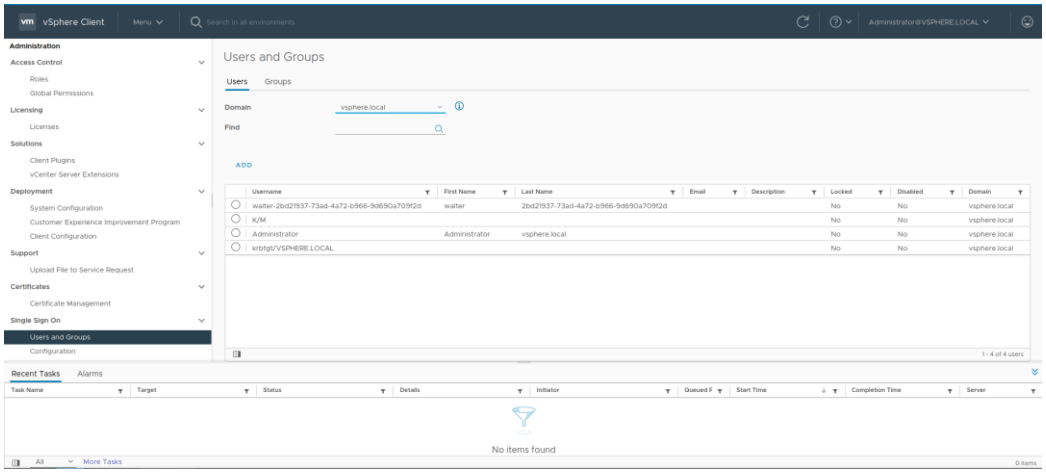
[24] Izvor: https://kb.vmware.com/s/article/1025279

6.4. Kreiranje korisnika i grupa te definiranje prava nad virtualnim mašinama

Kako bi korisnici imali pristup virtualnim mašinama, prvo ih moramo kreirati unutar administracijskog dijela vSphere sučelja. Korisnici koji se nalaze u lokalnoj bazi ESXi servera nisu vidljivi unutar vSphere sučelja kao što ni korisnici unutar vSphere-a nisu vidljivi na ESXi serveru.

6.4.1. Kreiranje korisnika i grupa

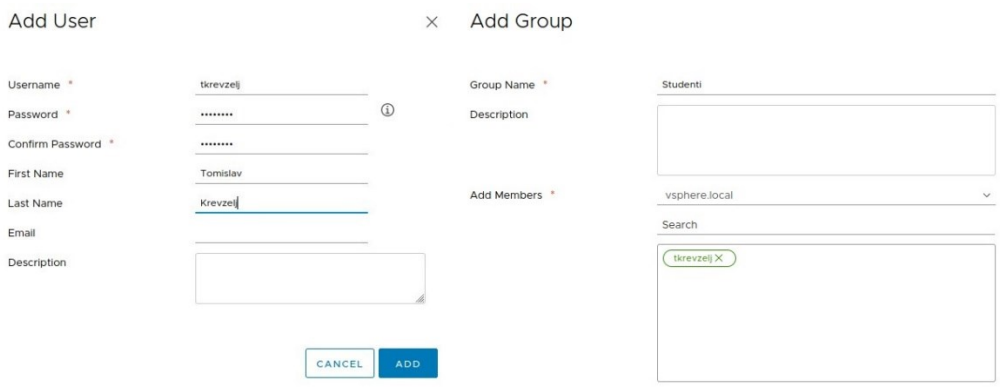
Upravljanju korisnicima i grupama možemo pristupiti otvaranjem glavnog menija te odabirom „*Administration*“ okvira. Potom unutar „*Single Sign On*“ odabiremo „*Users and Groups*“.



Slika 6.12 Prikaz korisnika i grupa unutar vSphere-a

Unutar odabranog prikaza možemo vidjeti trenutno kreirane korisnike i grupe te dodati nove odabirom gumba „*ADD*“. Kreirati ćemo korisnika imenom „Tomislav Krevzelj“ unutar domene „vsphere.local“ sa korisničkim imenom „tkrevzelj“ i lozinkom „Pa$$w0rd“.

Radi lakše administracije i dodjeljivanje prava, korisnike obično dodijelimo grupama, a u ovom slučaju ćemo kreirati grupu Studenti u koju ćemo smjestiti novokreiranog korisnika.



Slika 6.13 Kreiranje korisnika i grupa unutar vSphere-a

Iako su korisnici i grupe kreirani te se nalaze unutar vSphere-a, to ne znači da ujedno imaju i pravo pristupiti ili konfigurirati virtualne mašine koje se na vSphere-u nalaze. Ta prava ćemo dodijeliti pojedinačno na razini objekta, no prvo moramo kreirati rolu unutar koje ćemo definirati prava za korisnika, odnosno grupe u kojoj se korisnik nalazi.

6.4.2. Kreiranje role

Unutar vSphere-a već postoji nekoliko definiranih rola poput „*Administrator*“ role koja ima pravo modificirati bilo koju komponentu, „*Virtual Machine console user*“ koja ima pravo na interakciju s virtualnom mašinom ili „*No access*“ role koja nema pravo ni na što.

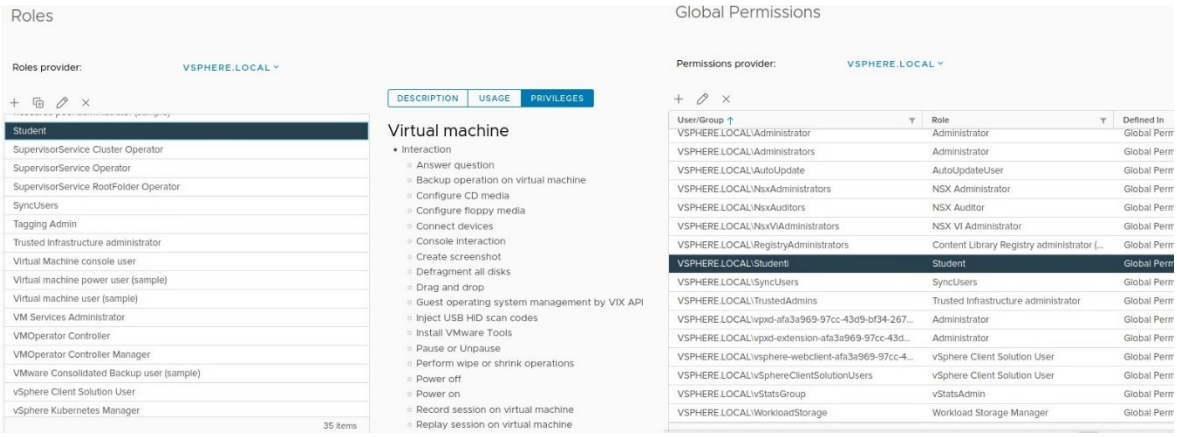
Ovdje ćemo dodatno kreirati rolu imena „Student“ koju ćemo dodijeliti svim korisnicima unutar grupe „Studenti“ te ćemo unutar nje omogućiti interakciju sa virtualnom mašinom te kreiranje snimke sustava.

To radimo na način da unutar istog administracijskog okvira odaberemo „*Roles*“ te potom odabirom „+“ znaka započnemo kreiranje nove role. Unutar „*Virtual Machine*“ okvira odabiremo „*Interaction*“ te „*Snapshot management*“.

Po završetku kreiranja role, istu ćemo dodijeliti na globalnoj razini za grupu „Studenti“. Na taj način svaki korisnik koji je član grupe „Studenti“ će imati prava na interakciju s virtualnom mašinom te upravljanje snimkama virtualne mašine.

Prava na globalnoj razini se definiraju u okviru „*Global Permissions*“ te odabirom „+“ znaka. Pod „*User/Group*“ odabirom „Studenti“, a za rolu biramo „Student“. Odabirom „*Propagate to children*“ otvaramo mogućnost dodjeljivanja prava na roditelj objektu (poput

direktorija u kojem se nalaze virtualne mašine) te će se ona automatski primijeniti na svu djecu unutar tog objekta (virtualne mašine u direktoriju).

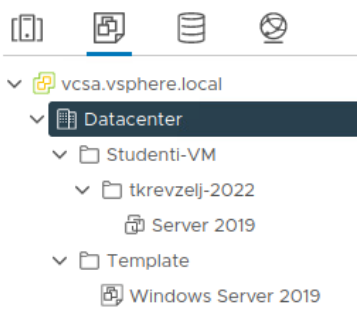


Slika 6.14 Rola Studenti te globalna dopuštenja role

Odabirom role te potom „*Privileges*“ okvira možemo provjeriti ispravnost konfiguracije, dok unutar „Global Permissions“ možemo provjeriti gdje se ta rola primjenjuje.

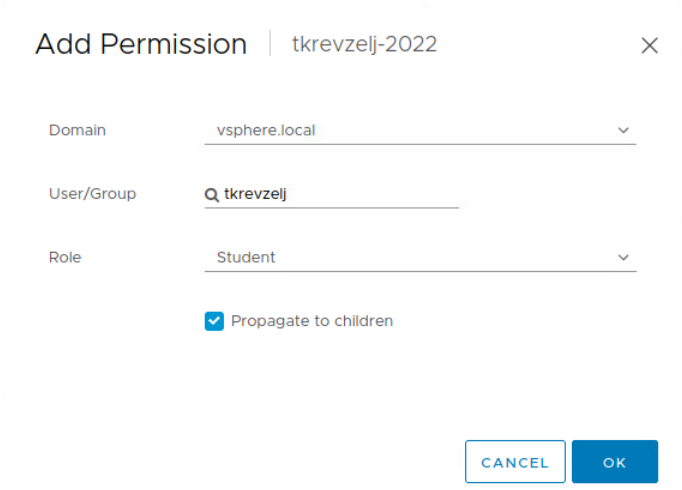
6.4.3. Dodjeljivanje prava nad virtualnom mašinom

Uz kreiranu virtualnu mašinu, korisnika, grupe i role, spremni smo dodijeliti prava nad virtualnom mašinom za kreiranog korisnika. No ipak, radi lakšeg snalaženja i administracije, dodatno ćemo kreirati direktorije za studentske virtualne mašine, za korisnika pojedinačno, te za predloške gdje nam se nalazi tzv. „*Golden image*“ za kreiranje dodatnih virtualnih mašina. Struktura i nazivi direktorija i virtualnih mašina prikazani su sljedećom slikom:



Slika 6.15 Struktura direktorija unutar vSphere-a

Ovom strukturom možemo definirati prava za više virtualnih mašina na razini direktorija za korisnika. U ovom primjeru ćemo korisniku „tkrevzelj“ dodijeliti rolu „Student“ za direktorij „tkrevzelj-2022“. Prava se dodjeljujemo na način da kliknemo na direktorij te pod „*Perimissions*“ okvirom odaberemo „+“ te za korisnika „tkrevzelj“ odaberemo rolu „Student“ te zakvačimo opciju „*Propagate to children*“ kako bi se prava odnosila na sve objekte unutar tog direktorija.



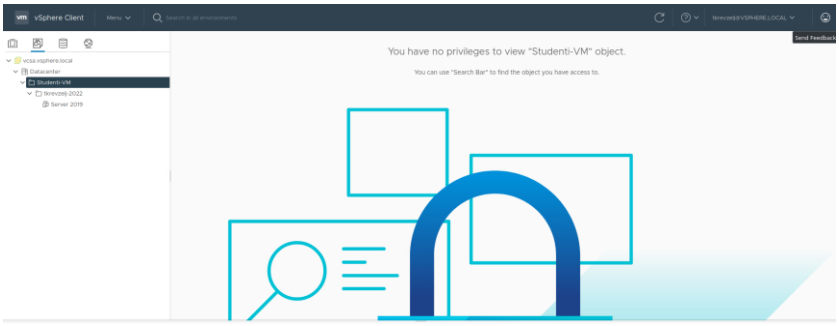
Slika 6.16 Dodjeljivanje prava nad direktorijem

Virtualne mašine smještene u tom direktoriju naslijediti će prava iz pripadajućeg direktorija, no treba imati na umu da eksplicitno zadana prava nad samim objektom imaju veći prioritet od onih naslijeđenih. Tako da ukoliko specifičnom korisniku zabranimo pristup nad virtualnom mašinom, a član je grupe koja ima pristup toj virtualnoj mašini, bez obzira što grupa ima pristup, tom korisniku biti će eksplicitno zabranjen te se neće moći koristiti tom virtualnom mašinom.

6.4.4. Provjera konfiguracije

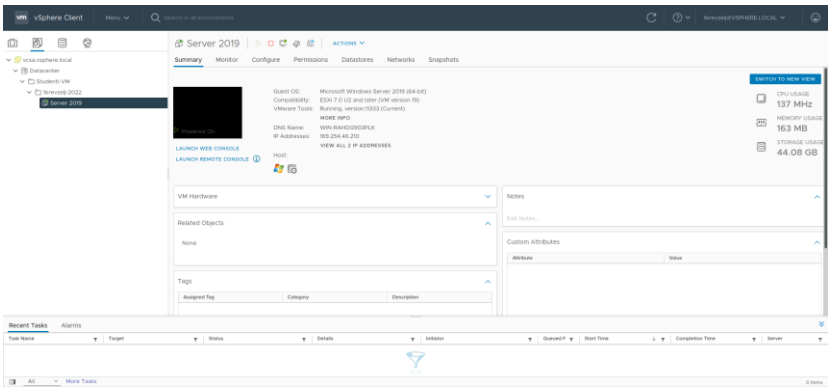
Ispravnost konfiguracije možemo provjeriti jednostavnim prebacivanjem s administratorskog korisnika na novokreiranog korisnika. Nakon prijave, korisnik bi trebao vidjeti „*You have no privileges to view XXX object*“, za sve objekte koji se ne nalaze u „tkrevzelj-2022“ direktoriju, dok bi unutar tog direktorija trebao imati pravo vidjeti,

pokrenuti te zaustaviti „Server 2019“ virtualnu mašinu. Također, trebao bi biti omogućen konzolni pristup, prikaz konfiguracije te osnovne funkcije za praćenje rada virtualne mašine i izrade snimki. Korisnik nema pravo mijenjati postavke, mijenjati konfiguraciju niti dodjeljivati dodatna prava unutar virtualne mašine. Uz to, ne može obrisati, niti dodati novu virtualnu mašinu. S obzirom da smo kreirali snimku naziva „Početno stanje VM“, korisnik se uvijek može vratiti na funkcionalno stanje u slučaju raspada virtualne mašine. Ukoliko korisnik pokuša pristupiti objektu nad kojim nema pristup, dočekati će ga sljedeća poruka:



Slika 6.17 Limitirana prava korisnika

U slučaju pristupanja objektu nad kojim ima pristup, može vidjeti sve pripadajuće funkcije i radnje koje su mu dostupne nad tim objektom.



Slika 6.18 Prava na upravljanje virtualnom mašinom

S obzirom da su prava definirana na razini direktorija, te ne postoje eksplicitna na razini virtualnih mašina, svaka virtualna mašina koja će se nalaziti u tom direktoriju biti će dostupna krajnjem korisniku na korištenje.

6.5. Kreiranje virtualnih mrežnih segmenata

Jedna od glavnih prednosti korištenja virtualne okoline je mogućnost logičkog odvajanja mrežnih segmenata u njoj. Logičkom segmentacijom mreže stvaramo novi sloj sigurnosti u kojem virtualnim mašinama omogućavamo pristup isključivo do željenih dijelova mreže. U većim organizacijama se to primarno dijeli na testnu i produkcijsku okolinu, no tih podjela može biti gotovo neograničeno.

Pojam koji vežemo uz logičku podjelu fizičke mreže je VLAN (eng. *Virtual LAN*). VLAN funkcionira na drugom sloj mrežnog stoga OSI modela kojeg nazivamo „Podatkovni sloj". Mrežni uređaji koji se nalaze unutar istog VLAN-a mogu međusobno komunicirati i izmjenjivati informacije, te ih također asociramo sa tzv. „*broadcast domain*" pojmom.

S obzirom da uređaji unutar iste *broadcast* domene mogu puno pričati, ponekad je potrebno izolirati te domene u vidu skaliranja mreže, smanjenja zagušenja prometa te općenito ubrzanog pronalaska ostalih uređaja na mreži. Organizacija koja definira pravila u mrežnoj komunikaciji uređaja IEEE[25] je definirala oznaku „VLAN ID" koja se dodaje na mrežni paket te pomoću koje možemo odrediti kojoj mreži taj paket pripada.

Virtualne mašine na vSphere-u koje su u različitim VLAN-ovima mogu komunicirati isključivo ako su međusobno povezane putem mrežnog usmjernika.

Sav mrežni promet, osim ako nije drukčije definirano, zadano putuje po VLAN-u 1. Nazivamo ga i „Nativnim VLAN-om". Oznaku VLAN-a možemo birati između 0 i 4095, što čini ukupno 4096 oznaka za odvajanje mrežnih segmenata. Treba uzeti u obzir da je VLAN 1 zadani te se ne koristi za kreiranje zasebnog mrežnog segmenta, VLAN 4095 predstavlja sve „*trunked*" VLAN-ove., dok VLAN 0 isključuje označavanje prometa na port grupi. U praksi to znači da ako imamo više virtualnih mašina, od kojih je jedna u VLAN-u 100, druga u VLAN-u 200, a treća u VLAN-u 4095; treća će ujedno biti član VLAN-a 100 te VLAN-a 200. U slučaju kada je potrebno koristiti više od 4094 VLAN-a, potrebno je proučiti privatne VLAN-ove, odnosno VLAN unutar VLAN-a. Takve konfiguracije primarno koriste Internet davatelji usluga, te za našu okolinu neće biti potrebni.

Komunikacija unutar vSphere-a možemo prezentirati na sličan način kao i kada je u pitanju fizička infrastruktura. Virtualne mašine spojene su na virtualne portove unutar virtualnog

[25] IEEE = Institute of Electrical and Electronics Engineers

preklopnika, te nad njima možemo primijeniti ista sigurnosna pravila kao i kada je u pitanju fizička oprema – poput definiranja veličine paketa, agregiranje izlaznih linkova, VLAN oznake, praćenja prometa, vezivanja MAC adresa i sl.

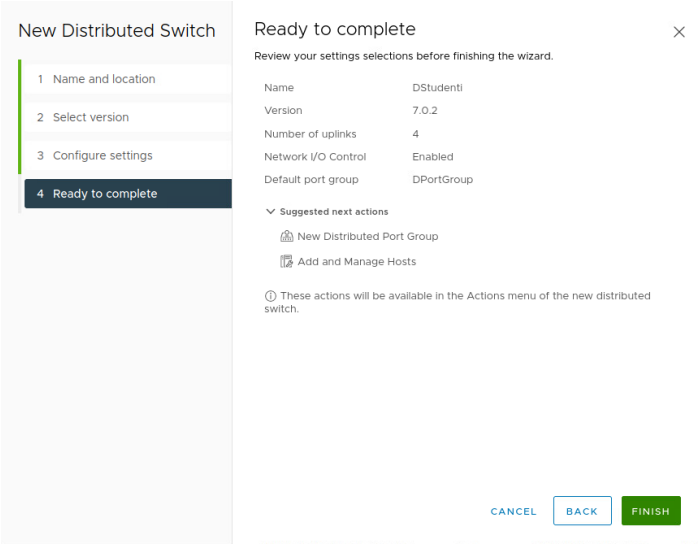
Za našu virtualnu okolinu, kreirati ćemo jedan virtualni preklopnik na kojem će se nalaziti nekolicina distribuiranih port grupa od kojih će svaka imati svoj VLAN ID. Ono što time postižemo je izolacija grupe virtualnih mašina od ostale grupe. Na taj način možemo koristiti iste IP postavke za isti set virtualnih mašina, npr. ako grupa 1 ima 3 servera sa adresama 10.0.0.1, 10.0.0.2 te 10.0.0.3, iste IP adrese možemo koristiti u drugoj grupi na druga tri servera. Ova funkcionalnost će se primarno doći do izražaja kada će se virtualne mašine raspoređivati putem Python aplikacije.

6.5.1. Kreiranje distribuiranog preklopnika

U slučaju kada želimo kreirati dodatne mrežne segmente, unutar vSphere sučelja moramo navigirati na „*Networking*“ odjel putem glavnog menija ili globus ikone u vSphere klijentu. Potom, unutar postojećeg podatkovnog centra desnim klikom odabiremo opciju „*Distributed Switch*“ pa „*New Distributed Switch*“. Kao što je već i ranije spomenuto, distribuirani preklopnici dostupni su svim ESXi serverima koji su dio tog podatkovnog centra, te ne zahtijevaju dodatnu konfiguraciju na ESXi serveru. Ono što je još bitno za napomenuti je da distribuirani preklopnici ovise o funkcionalnost vCenter servera za konfiguraciju. Kako je većina vCenter servera virtualizirana, u praksi to znači da kreiranjem neispravne konfiguracije VDS-a[26] koji rezultira padom vCenter servera, sve virtualne mašine koje su spojene na distribuirani preklopnik automatski gube vezu sa ostalim uređajima na mreži.

Naš distribuirani preklopnik zvat će se „DStudenti“, te ćemo unutar njega kreirati port grupe koju ćemo koristiti za komunikaciju virtualnih mašina unutar istog direktorija.

[26] VDS = Virtual Distributed Switch



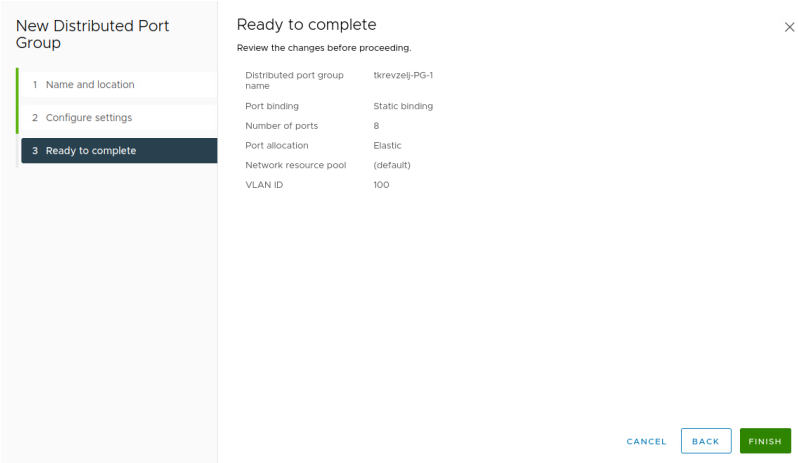
Slika 6.19 Kreiranje distribuiranog preklopnika

Distribuirani preklopnik će potom biti vidljiv u mrežnom dijelu vSphere-a te ga je moguće iskoristiti za kreiranje port grupa.

6.5.2. Kreiranje distribuirane port grupe

Unutar distribuiranog preklopnika potom kreiramo distribuiranu port grupu. Distribuirana port grupa je upravo kako joj i sam naziv kaže – grupa portova na virtualnom preklopniku. Na virtualne portove ćemo spajati virtualne mašine iz istog direktorija, a radi lakše administracije, VLAN ID možemo odrediti na port grupi, te će sve virtualne mašine unutar te grupe automatski imati dodijeljen odabrani VLAN ID.

Za primjer ćemo kreirati distribuiranu port grupu naziva „tkrevzelj-PG-1“ na način da desnim klikom odaberemo „DStudenti“ preklopnik te potom „*Distributed Port Group*“ i potom „*New Distributed Port Group*“. Broj portova može ostati proizvoljne veličine, a odabirom opcije „*Elastic*“ će se po potrebi povećati. Za VLAN opciju ćemo koristiti tip „VLAN“, a VLAN ID će biti bilo koji broj koji nije 0, 1 ili 4095 – u ovom primjeru biti će 100.



Slika 6.20 Kreiranje distribuirane port grupe

Po završetku, unutar distribuiranog preklopnika vidimo novokreiranu port grupu koja će se moći iskoristiti za međusobno povezivanje virtualnih mašina.

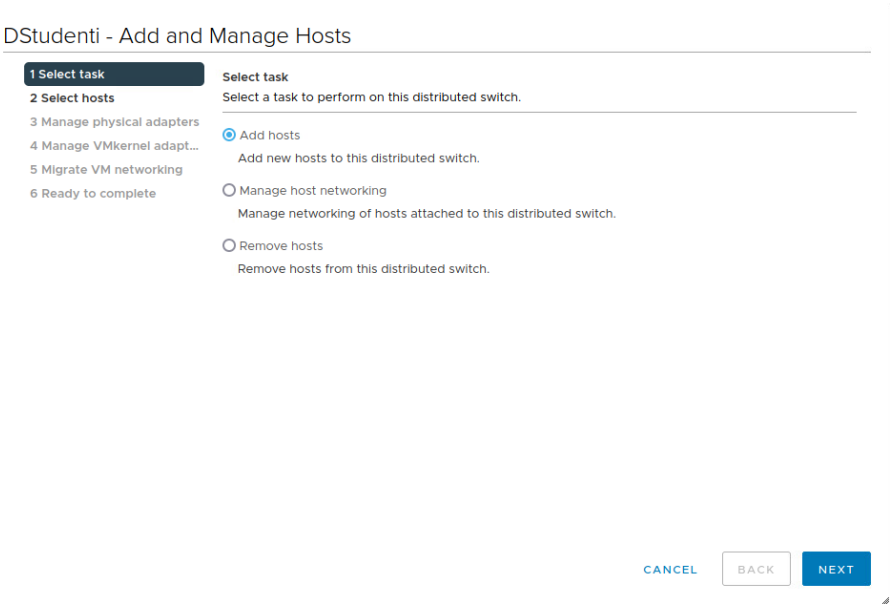
6.5.3. Dodavanje ESXi servera u distribuirani preklopnik

Kako bi mogli koristiti novokreirani distribuirani preklopnik, potrebno ga je dodati na ESXi servere. Servere dodajemo desnim klikom na preklopnik „DStudenti“ te odabirom „*Add and Manage Hosts*“ okvira.

Kako trenutno nema servera koji su članovi distribuiranog preklopnika, biramo opciju „*Add Hosts*“ te proširivanjem okvira „*New hosts*“ biramo „esxi1.vsphere.local“ te „esxi2.vsphere.local“.

Virtualne mašine koje će se nalaziti na ovom preklopniku neće imati izlaz na Internet, te im stoga nije potrebno dodijeliti izlaznu mrežnu karticu. Treba imati na umu da je ovdje lako moguće odabrati krivi izlazni port na mrežnoj kartici i odsjeći vezu koja je kreirana putem regularnog virtualnog preklopnika na vCenter server. U slučaju kada se vCenter server nalazi na tom ESXi serveru, gubitkom veze s ESXi-em gubimo i vezu na vCenter te je iz tog razloga preporučljivo čuvati snimku konfiguracije te imati standardni virtualni preklopnik pri ruci u slučaju pogrešne konfiguracije.

Ovisno o postojećoj konfiguracije, postoji mogućnost da će trebati preseliti virtualne mašine i VM kernel portove na ESXi-u na novu konfiguraciju, što u ovoj konfiguraciji nije slučaj.



Slika 6.21 Dodavanje ESXi servera na distribuirani preklopnik

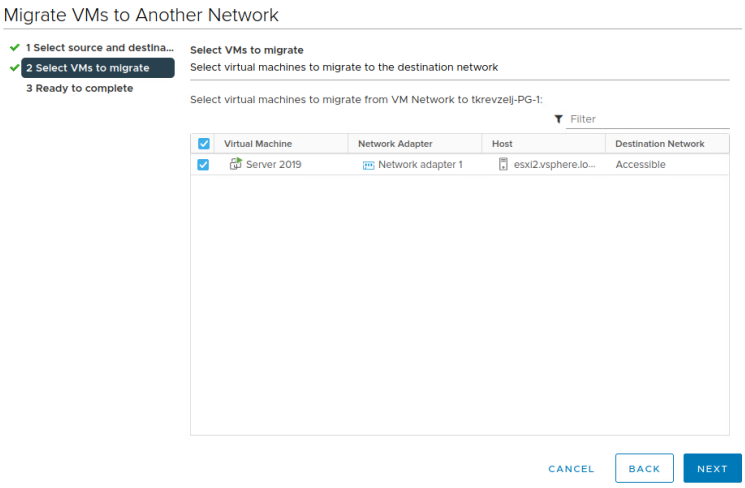
Nakon dodavanja ESXi servera, sve virtualne mašine koje se nalaze tom ESXi-u moći će komunicirati putem distribuiranog preklopnika.

6.5.4. Migriranje virtualnih mašina na distribuiranu port grupu

Popisu virtualnih mašina koji se nalaze unutar selektirane virtualne mreže možemo pristupiti odabirom „*VMs*“ okvira. Trenutno se u „tkrevzelj-PG-1“ ne nalazi niti jedna virtualna mašina, ali unutar „VM Network“ postoji „Server 2019“.

Da bi istu migrirali na novokreiranu port grupu, potrebno je desnim klikom na „VM Network“ odabrati opciju „*Migrate VMs to Another Network*“.

Za odredišnu mrežu odabrati ćemo „tkrevzelj-PG-1“, a virtualnu mašinu koju migriramo biti će „Server 2019“.



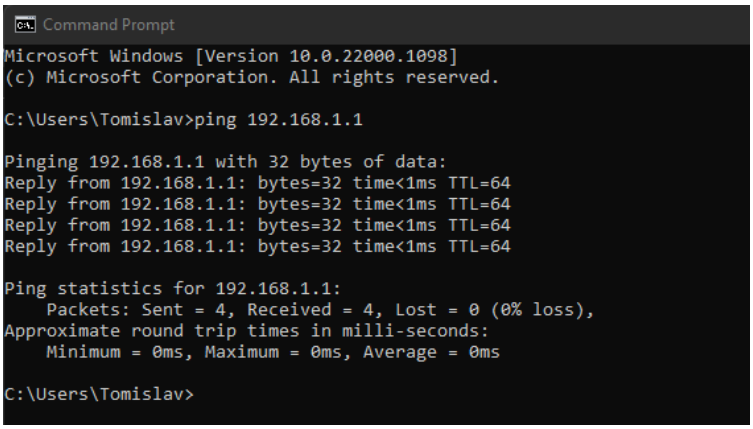
Slika 6.22 Migracija virtualnih mašina na drugu mrežu

Osim migracije jedne virtualne mašine, moguće je obostrano migrirati grupu virtualnih mašina između standardnog i distribuiranog preklopnika.

6.5.5. Testiranje povezanosti virtualnih mašina

S obzirom da virtualna mašina „Server 2019“ nema direktnu vezu s internetom, kreirati ćemo novu nazivom „Server 2019 – 2“ korištenjem „Windows Server 2019“ predloška kako je definirano u 6.1 poglavlju ovog rada. Ako novokreiranu virtualnu mašinu smjestimo u direktorij „tkrevzelj-2022“, automatski dodjeljujemo prava korisniku „tkrevzelj@vsphere.local“ za konzolni pristup i upravljanje snimkama virtualne mašine.

Ukoliko smo na obje virtualne mašine dodijelili IP adresu iz istog mrežnog raspona (npr. 192.168.1.0/24) te se obje nalaze unutar iste virtualne mreže (tkrevzelj-PG-1), pomoću *ping* naredbe možemo provjeriti povezanost virtualnih mašina.



Slika 6.23 Testiranje povezanosti virtualnih mašina korištenjem *ping* naredbe

Kako u trenutnoj topologiji nema mrežnog usmjernika, a komunikacija između različitih mrežnih segmenata nije moguća bez korištenja jednog, možemo biti sigurni da su obje virtualne mašine spojene na istu virtualnu mrežu unutar vSphere-a te da je migracija uspješno provedena za obje virtualne mašine. U slučaju kada se virtualne mašine nalaze na različitim mrežnim segmentima, bez obzira što je npr. na prvoj postavljena adresa 192.168.1.1/24, a na drugoj 192.168.1.2/24, neće moći međusobno komunicirati. Na taj način isti mrežni raspon možemo koristiti na više grupa virtualnih mašina, što rezultira mogućnošću stvaranja neovisnog testnog (ili produkcijskog) okruženja za sve korisnike.

7. Python aplikacija za upravljanje VMware okolinom

Python je dinamički programski jezik koji je visoko cijenjen zbog svoje jednostavnosti i čitljivosti. Također je jedan od najčešće korištenih programskih jezika na svijetu. Izvorno dizajniran od strane profesora Josepha Haroutuniana sa Sveučilišta Massachusetts u Bostonu, Python je besplatan softver otvorenog koda što znači da ga svatko može preuzeti i koristiti besplatno. Python programski jezik može se koristiti za izradu aplikacija namijenjenim krajnjim korisnicima ili poslužiteljima, te je neovisan o operacijskom sustavu na kojem se kod izvršava.

Od svoje kreacije, Python je postao jedan od najpopularnijih i najčešće korištenih programskih jezika u svijetu. Zbog svoje jednostavnosti prikladan je ujedno i početnicima i iskusnim programerima, a zbog objektno orijentiranog pristupa, omogućava projektiranje aplikacija kao skupinu objekata. Na taj način objekti se mogu programirati kako bi međusobno izmjenjivali informacije, za razliku od drugih programskih jezika gdje se kod orijentira na akciju koja se provodi.

Zahvaljujući jasnoj i jednostavnoj sintaksi i strukturi koda, programe je lako čitati i pisati., a uz brojne tečajeve dostupne na platformama poput Udemy, svatko može naučiti kodirati u Pythonu uz minimalan napor i poteškoće. Također, zbog svoje raširenosti, postoje brojni forumi i portali gdje je moguće postaviti pitanja i dobiti stručne odgovore na probleme koji se mogu pojaviti.

Za Python programski jezik postoje i brojni moduli, odnosno biblioteke funkcija. Unutar modula se nalazi kolekcija varijabli, funkcija i ostalih objekata koji se mogu pozvati iz glavnog programa, a koje možemo iskoristiti kako bi skratili količinu koda u glavnom programu.

Python aplikacija u ovom radu oslanjati će se na te module za upućivanje API zahtjeva prema vCenter Server poslužitelju te izvlačenje i formatiranje korisnika koji se nalaze unutar imeničkog sustavu na Windows Server poslužitelju.

7.1. Instalacija Python okruženja

Instalaciju Python okruženja provodimo na Linux Ubuntu operacijskom sustavu. U pravilu je programski jezik Python verzije 3 već instaliran na operacijskom sustavu, no u slučaju da nije, možemo ga instalirati korištenjem sljedećih naredbi:

```
sudo apt update
sudo apt install python3
sudo apt install python3-pip
```

Korištenjem prve naredbe ažurirati ćemo repozitorije, odnosno listu paketa koji su spremni za nadogradnju, dok ćemo korištenjem druge instalirati Python 3 programski jezik. Dodatno ćemo instalirati i „pip“ programski paket koji olakšava i ubrzava instalaciju modula za Python programski jezik. Korištenjem „pip“ naredbe možemo instalirati željene pakete, nadograditi postojeće, kao i deinstalirati nepotrebne. Jedna od posebnih funkcija ove naredbe koja ističe naspram ostalih je ta što možemo automatski instalirati i sve ovisne pakete koji su potrebni kako bi naš modul ispravno radio.

Kako vSphere verzija 7 u trenutku pisanja rada ne podržava kreiranje korisnika korištenjem Python API poziva, korisnike ćemo kreirati korištenjem „Go“ programskog jezika za što će nam poslužiti „govmomi“ biblioteka funkcija. Nju instaliramo korištenjem sljedeće naredbe:

```
curl -L -o -
"https://github.com/vmware/govmomi/releases/latest/download/g
ovc_$(uname -s)_$(uname -m).tar.gz" | tar -C /usr/local/bin -
xvzf - govc
```

Za spajanje na imenički sustav koristiti ćemo „python-ldap“ modul kojeg možemo instalirati korištenjem setup.py datoteke koju možemo preuzeti na https://github.com/python-ldap/python-ldap stranici ili korištenjem „pip“ paketa:

```
Python -m pip install python-ldap
```

Za upravljanje ostalim funkcijama poput kreiranja virtualnih mašina, promjena postavki te izrada snimki sustava, koristiti ćemo pyVmomi Python paket. Instaliramo ga korištenjem „pip“ naredbe:

```
pip install --upgrade pyvmomi
```

Svi gore navedeni paketi biti će neophodni za spajanje i konfiguriranje virtualne infrastrukture u VMware okruženju.

7.2. Povezivanje s imeničkim sustavom Windows Active Directory

Unutar imeničkog sustava Windows Active Directory kreirali smo 50 korisnika koje ćemo pomoću Python koda ubaciti unutar vCenter servera. Za početak se moramo spojiti sa Windows virtualnom mašinom korištenjem „ldap“ modula.

Provjeru korisničkog imena i lozinke smjestiti ćemo u *check_ldap_auth()* klasu unutar koje ćemo definirati parametre za spajanje na imenički sustav. Korištenjem odgovarajućeg filtera izvući ćemo popis korisnika iz Windows AD-a te ih spremiti u *results* polje te ga potom ispisati:

```
import ldap
def check_ldap_auth():
    """Verify credentials for username/password.
    Returns None on success or a string with error
description
    """
    LDAP_SRV = 'ldap://10.0.2.254'
    # fully qualified AD user name
    LDAP_USER = 'administrator@vsphere.local'
    LDAP_PWD = 'Pa$$w0rd'
    dn = 'OU=Studenti, DC=vsphere,DC=local'
    ldap_filter = 'objectClass=user'
    attrs = ['UserPrincipalName']
    try:
        ldap_cli = ldap.initialize(LDAP_SERVER)
        ldap_cli.set_option(ldap.OPT_REFERRALS, 0)
        ldap_cli.simple_bind_s(LDAP_USER, LDAP_PWD)
        print('Connected to LDAP Server ' + LDAP_SERVER)
    except ldap.INVALID_CREDENTIALS:
        ldap_client.unbind()
        print('Failed! Wrong username or password')
    except ldap.SERVER_DOWN:
        print('LDAP server is not available')
    # Connection successful, retrieving users from selected
path
```

```
        results = ldap_cli.search_s(dn, ldap.SCOPE_SUBTREE,
ldap_filter, attrs)
    results[0][1]['userPrincipalName'][0]
    Users = []
    for dn, entry in results:
        user = entry['userPrincipalName'][0]
        Users.append(user.decode("utf-8", "ignore"))

    for x in Users:
        print(x)
```

Kôd 7.1 Dohvaćanje korisnika iz imeničkog sustava

Programski kod rezultira ispisom korisnika u obliku korisnik@vsphere.local za svakog korisnika koji se nalazi u grupi „Studenti“ unutar vsphere.local domene. Svi rezultati se ispisuju u novi red.

U slučaju kada želimo korisnike koji se nalaze na određenoj godini, moramo prilagoditi filtar za odgovarajuću grupu. Korisnike s druge godine možemo filtrirati na sljedeći način:

```
    dn_2 = 'OU=Godina, DC=vsphere,DC=local'
    ldap_filter2 = '(&(objectClass=GROUP)(cn=2))'
    members = []
    result = ldap_cli.search_s(dn_2, ldap.SCOPE_SUBTREE,
ldap_filter2)
    if result:
        if len(result[0]) >= 2 and 'member' in result[0][1]:
            members_tmp = result[0][1]['member']
            for m in members_tmp:
                members.append(m.decode("utf-8", "ignore"))
    for m in members:
        results = ldap_client.search_s(m, ldap.SCOPE_SUBTREE,
ldap_filter, attrs)
        results[0][1]['userPrincipalName'][0]
        GrupaUsers = []
        for dn, entry in results:
            user = entry['userPrincipalName'][0]
            GrupaUsers.append(user.decode("utf-8", "ignore"))
        for h in GrupaUsers:
```

```
            print(h)
        print("\n")
    ldap_client.unbind()
```

Kôd 7.2 Filtriranje korisnika po grupi

Korisnike je moguće dohvatiti i korištenjem sljedeće PowerShell naredbe:

```
Get-ADGroupMember -Identity Studenti | Get-ADUser -Properties * | Select-Object GivenName, Surname, userPrincipalName
```

S obzirom da nam ispis korisnika ne znači mnogo, spremiti ćemo ga u CSV datoteku kako bi ju kasnije lakše iščitali i modificirali po potrebi. Tu datoteku potom možemo koristiti za kreiranje korisnika unutar vSphere-a.

7.3. Spremanje i iščitavanje korisnika iz CSV datoteke

Kako smo korisnike već dohvatili putem ladp modula, korištenjem csv modula iste možemo spremiti u Users.csv datoteku. Za kreiranje korisnika unutar vSphere-a potreban je naziv korisnika te lozinka, no dobro dođu i ime i prezime radi identifikacije korisnika. Za filtriranje korisnika potrebna nam je klasa u koju spremamo podatke od korisnika, kao i modificirani ldap upit u kojem dohvaćamo potrebne podatke:

```
    ldap_filter = 'objectClass=user'
    attrs = ['UserPrincipalName', 'givenName', 'sn']
    for dn, entry in results:
        userName = entry['givenName'][0]
        userSurname = entry['sn'][0]
        userPrincipalName = entry['userPrincipalName'][0]
        Users.append(userName.decode("utf-8", "ignore") + ','
+ userSurname.decode("utf-8", "ignore") + ',' +
userPrincipalName.decode("utf-8", "ignore"))
```

Kôd 7.3 Dohvaćanje korisnika u CSV formatu

Potom te korisnike možemo spremiti u klasu te u konačnici i u CSV datoteku za daljnju obradu.

```
import csv
class User:
    def __init__(self, givenName: str, surname: str,
userPrincipalName: str):
        self.givenName = givenName
        self.surname = surname
        self.userPrincipalName = userPrincipalName

    def __str__(self):
        return f"Korisnik: {self.givenName} {self.surname},
Logon: {self.userPrincipalName}"
with open('Users.csv', 'w') as csvFile:
    writer = csv.writer(csvFile)
    for x in Users:
        writer.writerow([x.givenName, x.surname,
x.userPrincipalName])
```

Kôd 7.4 Spremanje korisnika u CSV formatu

Uspješnost spremanja korisnika možemo provjeriti otvaranjem iste datoteke u načinu za čitanje (*eng. read*):

```
with open('Users.csv', 'r') as csvFile:
    reader = csv.reader(csvFile)
    Users = []
    for row in reader:
        Users.append(User(row[0], row[1], row[2]))
print("Users in CSV: ")
for x in Users:
    print(x)
```

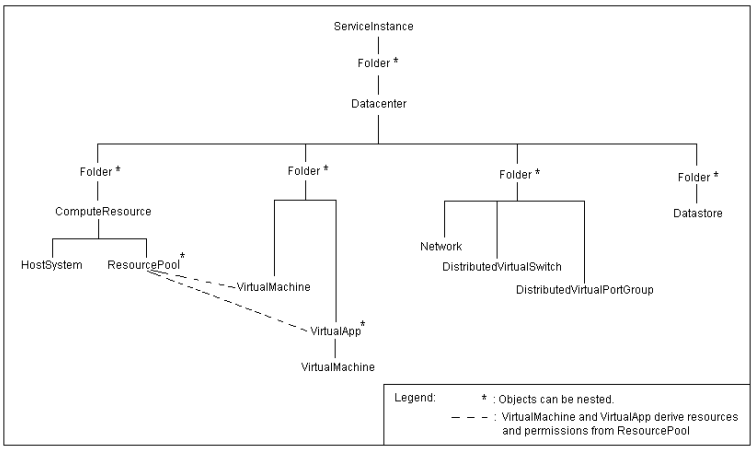
Kôd 7.5 Čitanje korisnika iz CSV datoteke

U slučaju uspješnog upisivanja podataka, CSV datoteka poprimiti će format oblika `Tomislav,Krevzelj,tkrevzelj@vsphere.local`, dok će prilikom iščitavanja rezultat ispisa biti `Korisnik: Tomislav Krevzelj, Logon: tkrevzelj@vsphere.local`.

7.4. Povezivanja na vSphere korištenjem pyVmomi modula

Python pVmomi modul je je set razvojnih alata koji pomoću vSphere API-a služi za upravljanje ESXi-em i vCenter serverom. Korištenjem modula za upravljanje vSphere-om imamo pristup isključivo do funkcija koje je VMware omogućio putem API-a. To znači da pristup upravljanju korisnicima nije moguć, s obzirom da se oni ne nalaze u popisu objekata vSphere API-a. Postoji nekoliko vrsti API-a, a pyVmomi koristi vSphere Web Services API čijoj dokumentaciji možemo pristupiti preko službenih VMware stranica. Iako postoje i drugi moduli, većina ih više nije održavana te se ne koristi u produkciji, dok se pyVmomi još uvijek aktivno održava.

Pristup objektima unutar vSphere-a definiran je samom strukturom vSphere-a. U trenutku kada želimo pristupiti objektu, moramo slijediti definiranu putanju za dohvaćanje, kreiranje i ažuriranje informacija o tom objektu. Struktura tih objekata prikazana je sljedećom slikom:



Slika 7.1 Struktura upravljanih objekata unutar vCenter hijerarhije[27]

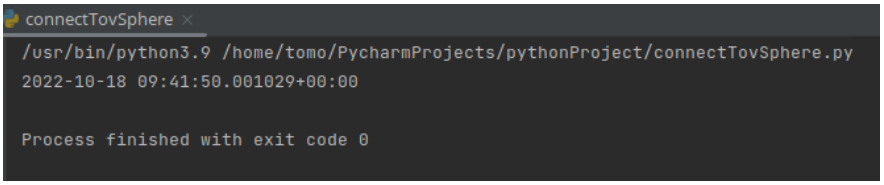
Za spajanje na vSphere koristimo SmartConnect funkciju koja je dostupna unutar pyVmomi modula. Ona prima informacije o adresi servera na koji se spajamo, korisniku i lozinki koju koristimo te portu ukoliko je različit od 443 (HTTPS port).

[27] Izvor: https://vdc-repo.vmware.com/vmwb-repository/dcr-public/6b586ed2-655c-49d9-9029-bc416323cb22/fa0b429a-a695-4c11-b7d2-2cbc284049dc/doc/vc_si_hierarchy.gif

```
from pyVim.connect import SmartConnect

si = SmartConnect(host="vcsa.vsphere.local",
user="administrator@vsphere.local", pwd='Pa$$w0rd')
```

Potom tu konekciju možemo iskoristiti za upravljanje objektima unutar vSphere-a, poput ispisa trenutnog vremena:

```
print(si.CurrentTime())
```



Slika 7.2 Podaci unutar CurrentTime objekta

Ako je konekcija bila uspješna, CurrentTime objekt bi trebao sadržavati, a print funkcija prikazati trenutno vrijeme postavljeno na vSphere serveru.

7.5. Dodavanje korisnika na vSphere

Korisnike koje smo spremili u CSV datoteku potom možemo dodati na vSphere. Kako Python modul „pyVmomi“ ne podržava kreiranje korisnika putem API poziva, u tu svrhu ćemo koristiti „govc“ modul te *environment* varijable. Za izradu korisnika studenata koristimo račun sa administratorskim ovlastima – u ovom slučaj administrator@vsphere.local. Korisnike potom dodajemo jednog po jednog, a u slučaju problema ispisujemo sadržaj povratne varijable. Kreiranje jednog korisnika možemo napraviti na sljedeći način:

```
import subprocess
import os


# GOVC has to be on $PATH
```

```
# env["PATH"] = "/usr/sbin:/sbin:" + env["PATH"]
def govc_linux(command):
    env = os.environ.copy()

    # Admin user to perform commands
    env["GOVC_USERNAME"] = "administrator@vsphere.local"
    env["GOVC_PASSWORD"] = "Pa$$w0rd"
    env["GOVC_URL"] = "https://vcsa.vsphere.local"
    env["GOVC_INSECURE"] = "true" #Allow insecure connetions
(invalid cert)

    process = subprocess.Popen(command, env=env, shell=True,
stdout=subprocess.PIPE, stderr=subprocess.PIPE)
    out, error = process.communicate()
    if out:
        print("Success: ")
        print(out)
    if error:
        print("Error: ")
        print(error)
    return out, error

# Test the connection
govc_linux("govc session.ls")


# New group and user info
newUserUsername = 'Baso'
newUserPassword = 'Pa$$w0rd'
newGroup = 'Studenti'

# Create a group, student, and assign that student to the
group
govc_linux("govc sso.group.create " + newGroup)
govc_linux("govc sso.user.create -p " + "'" + newUserPassword
+ "'" + " " + newUserUsername)
govc_linux("govc sso.group.update -a " + newUserUsername + "
" + newGroup)

# Test the output
output = govc_linux("govc sso.user.id " + newUserUsername)
```

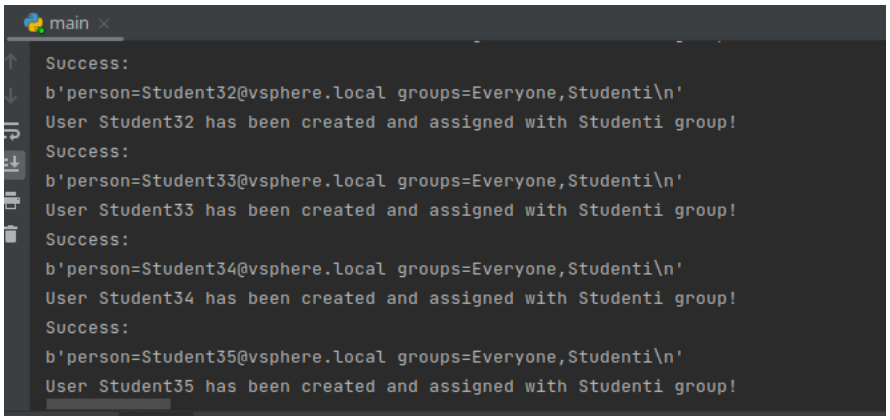
```
if output:
    print("User " + newUserUsername + " has been created and
assigned with " + newGroup + " group!")
else:
    print("Error occured")
```

Kôd 7.6 Kreiranje korisnika u vSphere-u putem GOVC naredbe

Dok za kreiranje više njih možemo učitati potrebne podatke iz CSV datoteke te unutar koda ubaciti imena i prezimena prilikom dodavanja korisnika na vSphere:

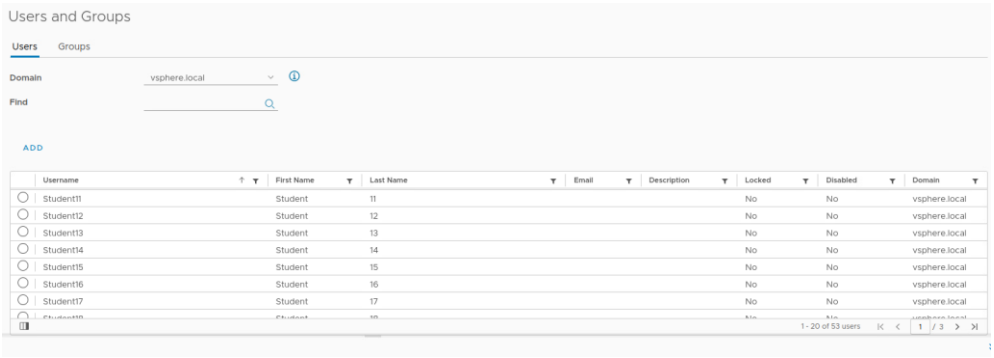
```
#govc sso.user.create -p 'Pa$$w0rd' -f 'givenName' -l 'Surname'
korisnik@vsphere.local

for x in Users:

    govc_linux("govc sso.user.create -p " + "'" + newUserPassword + "'" +
" -f " + x.givenName + " -l " + x.surname + " " + x.userPrincipalName)
```

Prilikom kreiranja korisnika Python aplikacija će prikazivati uspješnost kreiranja pojedinih korisnika.



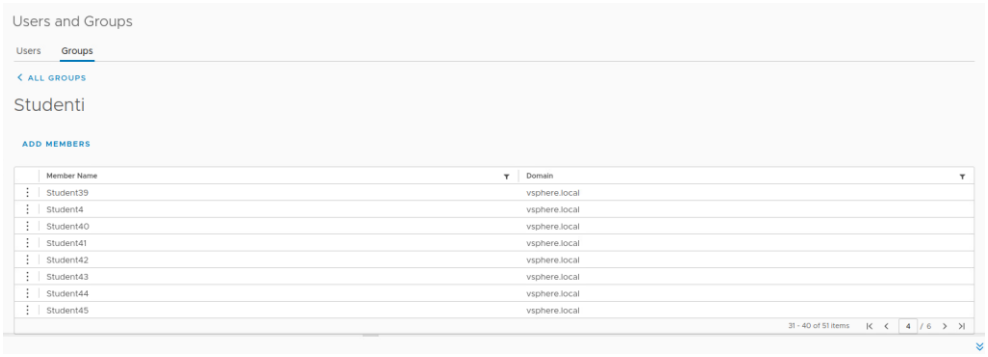
Slika 7.3 Ispis rezultata prilikom kreiranja korisnika

Te iste korisnike možemo pronaći i putem grafičkog sučelja vSphere-a unutar administracijskog dijela te odabirom „*Users and Groups*“ okvira.



Slika 7.4 Kreirani korisnici korištenjem Python aplikacije

S obzirom da su kreirani korisnici ujedno trebali biti pridruženi grupi, odabirom „*Groups*“ okvira te potom grupe „Studenti“ možemo utvrditi pripadnost novokreiranih korisnika.



Slika 7.5 Članovi grupe Studenti dodani putem Python aplikacije

Korisnike je moguće i ručno dodati putem „Add Members“ okvira, ali i ukloniti na isti način iz odabrane grupe.

7.6. Kreiranje virtualne mašine

Po uspješnom spajanju na vCenter, imamo mogućnost kreiranja virtualnih mašina sa željenim parametrima. Ukoliko proučimo dokumentaciju, vidimo da je za izradu virtualne mašine potrebno poslati nekoliko informacija kako bi se ona uspješno kreirala. Informacije poput naziva virtualne mašine, vrste operacijskog sustava, količine procesora i radne memorije neophodne su za izradu virtualne mašine korištenjem API-a. Uz to, potrebno je

definirati lokaciju datoteka virtualne mašine, kao i ESXi server na kojem će se ta virtualna mašina nalaziti.

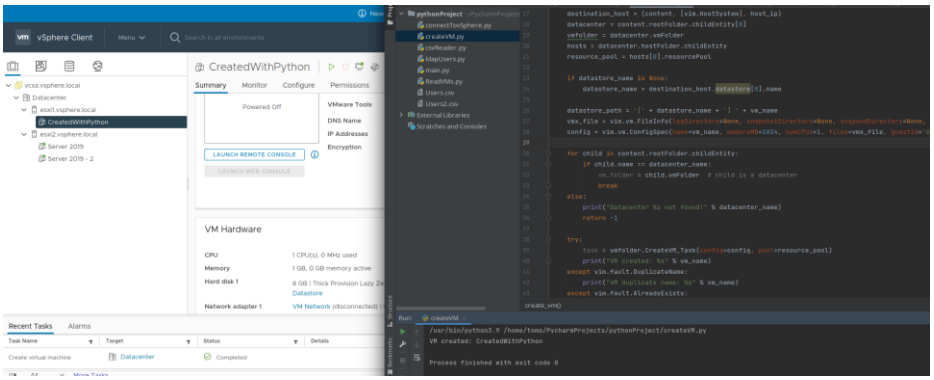
Virtualna mašina može primiti i mnogo više parametara, a za modificiranje istih potrebno je konzultirati se sa VMware dokumentacijom.

```
from pyVim.connect import SmartConnect, Disconnect
from pyVmomi import vim, vmodl


# vCenter server settings
IP = 'vcsa.vsphere.local'
USER = 'administrator@vsphere.local'
PWD = 'Pa$$w0rd'
PORT = 443


def create_vm(si, vm_name, datacenter_name, host_ip,
datastore_name=None):

    content = si.RetrieveContent()
    destination_host = (content, [vim.HostSystem], host_ip)
    datacenter = content.rootFolder.childEntity[0]
    vmfolder = datacenter.vmFolder
    hosts = datacenter.hostFolder.childEntity
    resource_pool = hosts[0].resourcePool

    if datastore_name is None:
        datastore_name = destination_host.datastore[0].name

    datastore_path = '[' + datastore_name + '] ' + vm_name
    vmx_file = vim.vm.FileInfo(logDirectory=None,
snapshotDirectory=None, suspendDirectory=None,
vmPathName=datastore_path)
    config = vim.vm.ConfigSpec(name=vm_name, memoryMB=1024,
numCPUs=1, files=vmx_file, guestId='Other', version='Other')

    for child in content.rootFolder.childEntity:
        if child.name == datacenter_name:
```

```
            vm_folder = child.vmFolder  # child is a
datacenter
            break
    else:
        print("Datacenter %s has not been found!" %
datacenter_name)
        return -1

    try:
        task = vmfolder.CreateVM_Task(config=config,
pool=resource_pool)
        print("VM created: %s" % vm_name)
    except vim.fault.DuplicateName:
        print("VM duplicate found: %s" % vm_name)
    except vim.fault.AlreadyExists:
        print("VM name %s already exists in datacenter." %
vm_name)


def main():
    si = SmartConnect(host=IP, user=USER, pwd=PWD,
port=int(PORT))

    try:
        content = si.RetrieveContent()
        create_vm(si, "CreatedWithPython", "Datacenter",
"esxi1.vsphere.local", "Datastore")

    except vmodl.MethodFault as error:
        print("Unable to connect to vSphere server." +
error.msg)
        return -1

    return 0


if __name__ == '__main__':
    main()
```

Kôd 7.7 Program za kreiranje virtualne mašine sa željenim parametrima

Uspješno kreiranje virtualne mašine možemo provjeriti prijavom u vSphere sučelje unutar *Hosts and Clusters* okvira.



Slika 7.6 Virtualna mašina kreirana korištenjem pyVmomi modula

Kreiranjem virtualnih mašina putem aplikacije zahtjeva pojedinačno navođenje resursa kojom će se ta virtualna mašina koristiti. Korištenjem grafičkog alata, vSphere će automatski odabrati resurse i popuniti vrijednosti ovisno o odabranom operacijskom sustavu, dok za API pristup neće. Tako je moguće kreirati virtualne mašine bez diskova i mreža, koje u praksi nije moguće koristiti, no mogu poslužiti za testiranje funkcionalnosti sustava.

7.7. Ispisivanje virtualnih mašina

Virtualne mašine također možemo ispisati korištenjem pyVmomi modula. Potrebno je autenticirati se na vSphere te pročitati summary objekt virtualne mašine. S obzirom da nismo sigurni gdje se sve virtualne mašine nalaze, ispisu pristupamo rekurzivno – ulazimo u direktorije i ispisujemo VM sve dok postoje objekti tog tipa. Ispisujemo informacije o nazivu virtualne mašine, vrsti operacijskog sustava, količini procesora i radne memorije. Ispisujemo i stanje virtualne mašine (uključena/isključena/u hibernaciji), te IP adresu. Također uključujemo i informaciju o tome da li je virtualna mašina u obliku predloška.

```
from pyVim.connect import SmartConnect, Disconnect
from pyVmomi import vim, vmodl
```

```
# vCenter server settings
IP = 'vcsa.vsphere.local'
USER = 'administrator@vsphere.local'
PWD = 'Pa$$w0rd'
PORT = 443


def print_vm(virtual_machine):
    # Print VM info from object summary
    summary = virtual_machine.summary

    print("Name         : ", summary.config.name)
    print("Guest        : ", summary.config.guestFullName)
    print("CPU          : ", summary.config.numCpu)
    print("Memory       : ", summary.config.memorySizeMB)
    print("Is Template? : ", summary.config.template)
    print("Path         : ", summary.config.vmPathName)
    print("State        : ", summary.runtime.powerState)
    if summary.guest.ipAddress is not None:
        print("IP           : ", summary.guest.ipAddress)
    else:
        print("IP           : None")
    print("")


def main():
    si = SmartConnect(host=IP, user=USER, pwd=PWD,
port=int(PORT))

    try:
        content = si.RetrieveContent()

        root = content.rootFolder  # searching from root
folder
        object_type = [vim.VirtualMachine]  # searching for
VM type of object
        recursive = True  # search as deep as needed
        root_object =
content.viewManager.CreateContainerView(
            root, object_type, recursive)
```

```
            children = root_object.view
            for child in children:
                print_vm(child)

        except vmodl.MethodFault as error:
            print("Unable to connect to vSphere server." +
error.msg)
            return -1

        return 0


if __name__ == '__main__':
    main()
```

Kôd 7.8 Ispisivanje virtualnih mašina korištenjem pyVmomi modula

Unutar objekta Summary nalaze se i druge informacije, npr. UUID (vCenter specifičan ID za virtualnu mašinu), broj mrežnih adaptera ili količina rezervirane radne memorije. Za popis svih dostupnih objekata potrebno je pristupiti VMware repozitoriju na njihovim službenim stranicama.

7.8. Ispis i izrada snimke sustava

Snimku sustava moguće je odraditi kroz Python API ukoliko ustupimo potrebne parametre. Kao i u ostalim slučajevima, potrebno je autenticirati se korištenjem SmartConnect funkcije te potom definirati putanju ili rekurzivno proći kroz direktoriji za prikaz kreiranih snimki virtualnih mašina.

```
def get_snapshots(snapshots, snapshot_path):
    snapshot_paths = []

    if not snapshots:
        return snapshot_paths

    for snapshot in snapshots:
```

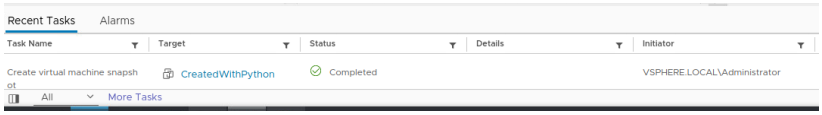
```
            if snapshot_path:
                this_snapshot_path = snapshot_path + '/' +
snapshot.name
            else:
                this_snapshot_path = snapshot.name

            snapshot_paths.append(this_snapshot_path)
            snapshot_paths = snapshot_paths +
get_snapshots_recursively(snapshot.childSnapshotList,
this_snapshot_path)

        return snapshot_paths
```

Kôd 7.9 Program za pronalaženje kreiranih snimki nad virtualnim mašinama

U slučaju kada želimo izraditi snimku sustava, sve što nam je potrebno je naziv virtualne mašine i željeni naziv snimke:

```
vm_name = "Server 2019"
snapshot_name = "Python snapshot"
description = "Created with pyVmomi module"
vm = get_obj(content, [vim.VirtualMachine], vm_name)
invoke_and_track(vm.CreateSnapshot(snapshot_name, description, memory =
False, quiesce=False))
```

Uspješnost provođenja funkcije za izradu snimki možemo provjeriti u nedavnim zadacima unutar vSphere sučelja:



Slika 7.7 Kreiranje snimke sustava pomoću pyVmomi modula

Također, unutar direktorija gdje je virtualna mašina pohranjena, pojaviti će se nove datoteke, a nova delta datoteku možemo identificirati po nazivu i formatu „CreatedWithPython-000001.vmdk“.

7.9. Kloniranje virtualne mašine

Virtualne mašine kloniramo koristeći predložak virtualne mašine uz dodjelu novog naziva i putanje. Naravno, kao i u prethodnim primjerima, potrebno je uspostaviti vezu na vCenter, locirati direktorij i virtualnu mašinu koju kloniramo. Unutar pyVmomi modula postoji funkcija search_for_obj kojoj možemo specificirati da traži virtualnu mašinu tipa predložak.

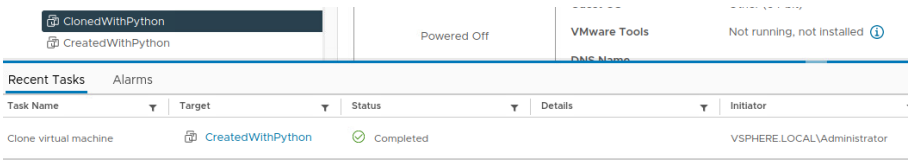
```
templateVM = pchelper.search_for_obj(content, [vim.VirtualMachine], „Windows Server 2019“, vm_folder)
```

Potom taj predložak koristimo za izradu nove virtualne mašine:

```
templateVM = "Server 2019"
vm_name = "ClonedVMWithPython"
vm_folder = datacenter.vmFolder
def clone_vm(si, templateVM, vm_name, vm_folder, location):
    clone_settings = vim.vm.CloneSpec(powerOn=True,
template=False, location=location,
snapshot=template.snapshot.rootSnapshotList[0].snapshot)
#Cloned specifications with removed template tag
    task = template.Clone(name=vm_name, folder=vm_folder,
spec=clone_settings)
    print("Success! Created new cloned VM: %s ", vm_name)
```

Kôd 7.10 Kloniranje virtualne mašine

Pri uspješnom izvršavanju pojavljuje se i zapis unutar vSphere sučelja:



Slika 7.8 Rezultat kloniranja virtualne mašine

Ako pristupimo pohrani, možemo primijetiti da je kreiran novi direktorij sa datotekama koje su veličinom identične onima iz izvornog direktorija, no drukčijeg naziva.

7.10. Kreiranje virtualnog preklopnika i port grupe

Virtualne preklopnike i pripadajuće port grupe možemo konfigurirati pute pyVmomi modula, no treba voditi računa o tome da prilikom dodavanja ESXi servera ne smijemo pregaziti postojeću mrežnu konfiguraciju. Za kreiranje virtualnog preklopnika potreban nam je naziv, broj portova za virtualne mašine te izlazni port na mrežnoj kartici.

```
content = si.RetrieveContent()
server = get_obj(content, [vim.HostSystem],
"esxi1.vsphere.local")
network = server.configManager.networkSystem
vs_name = "vSwitchPython"
num_ports = 16
nic_name = "vmnic1"

def create_vswitch(network, vs_name, num_ports, nic_name):
    vs_spec = vim.host.VirtualSwitch.Specification()
    vs_spec.numPorts = num_ports
    vss_spec.bridge =
vim.host.VirtualSwitch.BondBridge(nicDevice=[nic_name])
    network.AddVirtualSwitch(vswitchName=vs_name,
spec=vs_spec)
```

Kôd 7.11 Kreiranje virtualnog preklopnika

Potom je potrebno kreirati port grupu koje ćemo dodijeliti tom preklopniku:

```
def create_pg(network, pg_name, vs_name):
    pg_spec = vim.host.PortGroup.Specification()
    pg.name = pg_name
    pg.vlanId = 100
    pg.vswitchName = vss_name

    sec_policy = vim.host.NetworkPolicy.SecurityPolicy()
    sec_policy.allowPromiscuous = True
    sec_policy.forgedTransmits = True
    sec_policy.macChanges = False
```

```
pg.policy = vim.host.NetworkPolicy(security=sec_policy)

host_network_system.AddPortGroup(portgrp=pg_spec)
```

Kôd 7.12 Kreiranje port grupe sa VLAN ID 100

Treba voditi računa da virtualne preklopnik neće biti vidljiv na ESXi serveru na kojem nije dodijeljen. S obzirom da je ovdje riječ o klasičnom virtualnom preklopniku, potrebno ga je kreirati na svim ESXi serverima unutar lokalne domene.

7.11. Kreiranje direktorija

Direktorije unutar vSphere-a kreiramo pozivanjem CreateFolder funkcije kojoj je potrebno proslijediti dva parametra – lokaciju i naziv.

```
def create_folder(host_folder, folder_name):
    host_folder.CreateFolder(folder_name)

datacenter = get_obj(content, [vim.Datacenter], "Datacenter")
#Get Datacenter object
create_folder(datacenter.vmFolder, "Novi direktorij")
print("Folder has been created successfuly")
```

Kôd 7.13 Kreiranje direktorija za VM

Kreirane direktorije potom možemo koristiti isključivo za smještaj virtualnih mašina. U slučaju kada je potrebno smjestiti ESXi servere unutar tog direktorija, koristimo *hostFolder* parametar.

7.12. Rekonfiguracija virtualne mašine

Virtualne mašine je moguće naknadno konfigurirati na sličan način na kojih ih i kreiramo. Za potrebe rekonfiguracije pozivamo Reconfig_VM funkciju unutar koje šaljemo novu

konfiguraciju virtualne mašine. Ako želimo omogućiti dodavanje radne memorije i procesora u radu, potrebno je unutar ConfigSpec promijeniti vrijednost na True (početna vrijednost je postavljena na False).

```
config = vim.vm.ConfigSpec()

config.numCPUs = 2
config.cpuHotRemoveEnabled = True
config.cpuHotAddEnabled = True

config.memoryMB = 2048
config.memoryHotAddEnabled = True


task = vm.ReconfigVM_Task(spec=config)
```

Kôd 7.14 Rekonfiguracija postavki virtualne mašine

Rekonfiguracijom možemo promijeniti trenutne, dodati nove ili ukloniti stare parametre virtualne mašine. Svi dostupni parametri nalaze se u *vSphere Web Services API* dokumentaciji, pri čemu treba odabrati upravljani objekt oblika *VirtualMachine*.

8. Testiranje funkcionalnosti aplikacije

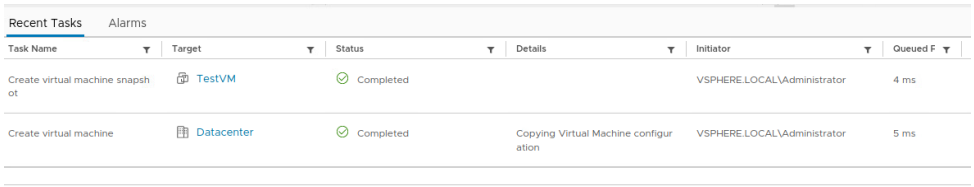
Za potrebe provođenja testiranja aplikacije koristiti ćemo jednostavnu petlju koja će redom pozivati funkcije za povlačenje i dodavanje korisnika, izradu virtualne mašine te snimke te virtualne mašine. S obzirom da ne možemo upravljati pravima korisnika, te smo limitirani diskovnim prostorom, kreirati ćemo samo jednu inačicu virtualne mašine. Na njoj je dodatno moguće definirati ESXi server po izboru, postavke poput broja jezgri i radne memorije, te dodijeliti virtualnu mrežu po izboru. Po završetku se ispisuju podaci virtualne mašine te se nudi mogućnost kreiranja snimke i pokretanja virtualne mašine.

U tu svrhu spremamo niz varijabli poput USER, PWD, VM_NAME, CPU_COUNT, RAM_MEM, NET_TYPE, TARGET itd.

Te parametre potom prosljeđujemo funkcijama koje smo do sada kreirali:

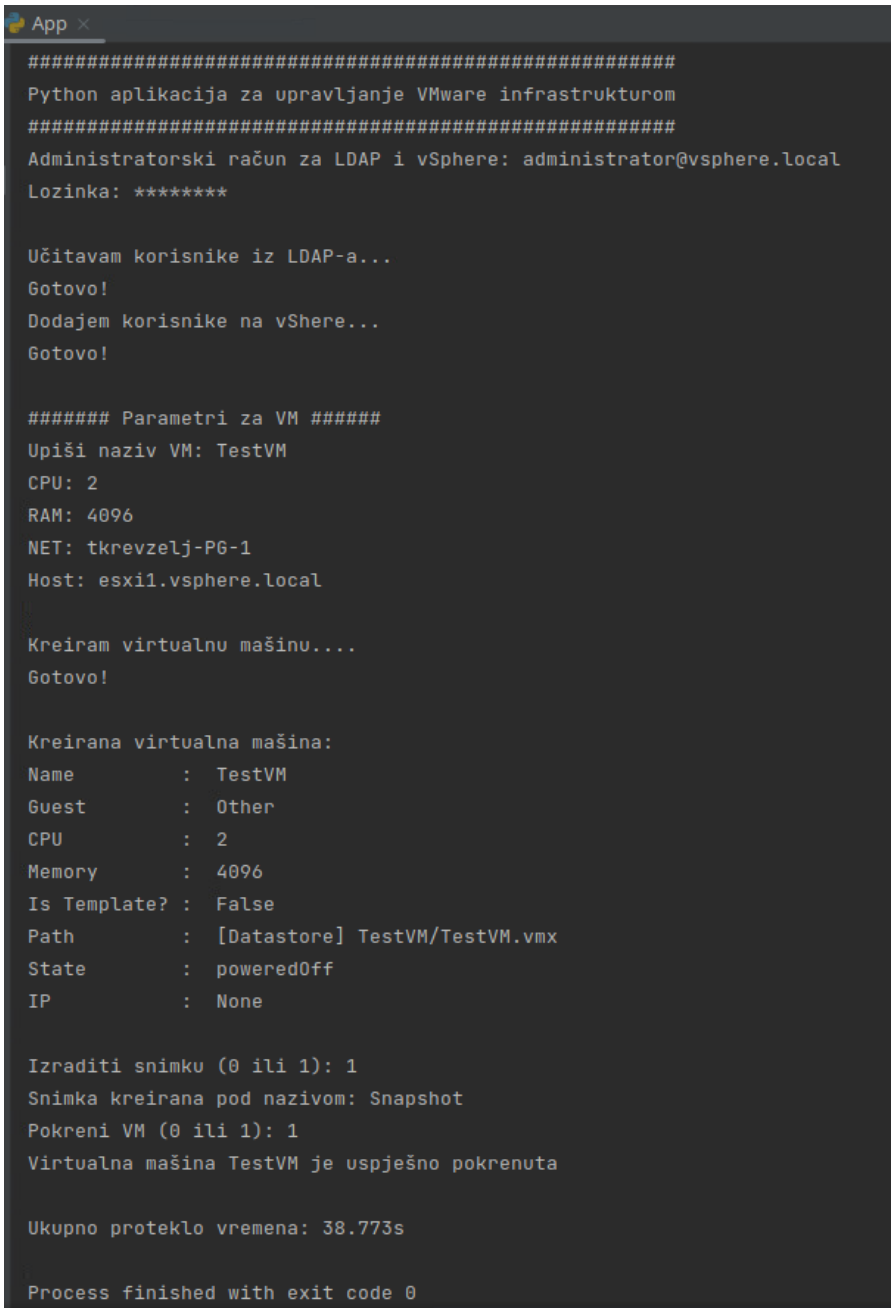
connectTovSphere.py, createVM.py, csvReader.py, csvWriter.py, ldapUsers.py, ReadVMs.py, ChangeVM.py

Sve funkcije se pozivaju iz main.py programa te su varijable globalno dostupne. Korisnici se spremaju i iščitavaju iz Users.csv datoteke. Dodatno, bilježimo vrijeme koje je proteklo od pokretanja programa do završetka programa.



Slika 8.1 Ispis rada aplikacije unutar vSphere sučelja

vSphere će ispisivati trenutne radnje aplikacije unutar *Recent Tasks* okvira. Unutar tog okvira nećemo vidjeti kreiranje korisnika i grupa, niti dodavanje korisnika u grupu, no postojat će zapis o kreiranju i rekonfiguraciji virtualne mašine, kao i o kreiranju snimke sustava.



Slika 8.2 Testiranje funkcionalnosti aplikacije

Unutar same aplikacije možemo birati što će sve biti prikazano i na koji način. Jedna od prednosti korištenja Python-a naspram PowerShell-a je ta što ispis možemo formatirati i prezentirati na nama odgovarajući način – u obliku teksta ili slike, tablice ili neke vanjske datoteke. Izradom grafičkog sučelja cijeli proces se može dodatno ubrzati, a kreiranjem novih ulaznih parametara moguće ga je i u potpunosti skriptirati.

Zaključak

S eksponencijalnim rastom računalnih resursa, sve je veći broj aplikacija i servisa koji se smještaju u virtualna računala. Ovisno o prirodi i svrsi tih aplikacija te financijskim mogućnostima kompanije, iste će se smjestiti na lokalnoj ili udaljenoj infrastrukturi. Održavanje infrastrukture na kojoj se te aplikacije pokreću nije komplicirano kada je u pitanju par servera sa svega par virtualnih mašina, no može postati itekako mukotrpno kada se njihov broj popne na desetke servera i stotine virtualnih mašina. Ručna (re)konfiguracija takve infrastrukture oduzela bi sate za nekolicinu stručnjaka IT odjela, što je u najmanju ruku neefikasno i besmisleno jer postoji velika šansa da se dio preskoči ili neispravno konfigurira. Iako su se prvi oblici automatizacije pojavili još davno prije doba klasičnih računala, glavna svrha ostala je nepromijenjena - zamijeniti čovjeka robotom tamo gdje je to moguće. U računalnom svijetu to predstavlja skriptu ili program koji izvršava zadatak uz minimalnu interakciju čovjeka. U virtualizacijskom svijetu kreiramo skripte i mini programe koji će se spojiti na virtualizacijski server i odraditi niz zadataka umjesto nas.

Automatizacija vSphere-a nipošto nije novi pojam. Već dugi niz godina VMware održava i redovito nadograđuje PowerCLI - PowerShell modul za upravljanje VMware infrastrukturom koji omogućava automatizaciju svih aspekata samog vSphere-a. Dok je PowerCLI stari, API spada u noviji način programiranja infrastrukture te je dostupan na nizu programskih jezika – poput Python-a, Perl-a, Ruby-a ili Jave.

PowerCLI predstavlja programsko sučelje koje se sastoji od kolekcije PowerShell modula koje nude niz naredbi za upravljanje VMware proizvodima. Osim vSphere-a, moguće ga je iskoristiti za konfiguraciju VMware Horizon-a, VMware NSX-T-a ili VMware Cloud Director-a (i ostalih). Konfiguracije putem programskih jezika koje se oslanjaju na API pozive najčešće se odnose na jedan proizvod.

Do prije par godina nazivati PowerShell programskim jezikom izazivalo bi podsmjehe i odmahivanje glavom, dok danas to više nije slučaj. Iako ga i dalje klasificiramo kao jezik za skriptiranje, PowerShell danas nudi pregršt funkcija koje spadaju u kategoriju programskog jezika. Ako je za programski jezik preduvjet da ima kompajler, PowerShell ga nudi – moguće je skripte pretvoriti u .exe datoteke; ako je preduvjet da je više objektno orijentiran, onda u istu kategoriju spadaju i C, Go ili Rust; ako je preduvjet da ima statičke tipove podataka, onda ni Python nije programski jezik; ako je preduvjet da se mogu izraditi grafičke aplikacije

– i to postoji u obliku .NET API-a. PowerShell možda neće biti najbolji izbor u slučaju da idemo pisati vlastiti kompajler ili programirati web server, ali isto tako nećemo ni koristiti Javu za upravljanje memorijom ili C++ za premještanje Windows datoteka. U konačnici se sve svodi na tip i kompleksnost zadatka kojeg pokušavamo obaviti korištenjem skripti ili aplikacije.

Python također možemo klasificirati kao skriptni jezik. Ono u čemu je Python u prednosti naspram PowerShell-a je činjenica da ga je lakše migrirati s jednog operacijskog sustava na drugi jer je podržan na gotovo svim operacijskim sustavima, dok u slučaju PowerShell-a to nije slučaj. S obzirom da je Python zamišljen kao multiplatformski programski jezik, ima jednu od najvećih baza korisnika koja je svakim danom sve veća i veća. Zbog svoje čitljivosti i jednostavne sintakse, također je jedan od prvih jezika koji se uče prilikom ulaska u svijet programiranja. Problematični dijelovi programiranja u Pythonu najviše se pojavljuju prilikom korištenja vanjskih modula za izradu aplikacije. Postoji velika količina modula koja nije ispravno dokumentirana, nema testne scenarije ili jednostavno nije ažurirana godinama, a dandanas je dostupna za preuzimanje i korištenje unutar programskog koda. Takvi moduli mogu rezultirati nekompatibilnošću koda s novijim verzijama, pogotovo prilikom migracije Python koda s verzije 2 na verziju 3.

Najveća prednost aplikacije koja se oslanja na API pozive je ta što nismo vezani uz specifičan programski jezik. S obzirom da je API komponenta koja omogućava različitim platformama, aplikacijama i sistemima da se spoje, čitaju ili promijene informacije, više ne ovisimo o poznavanju i korištenju PowerShell koda, jer istu funkcionalnost možemo postići na više različitih načina. Ono što je problematično kod ovog tipa pristupa je činjenica da je API dinamičan, te se dohvaćanje ili upisivanje nekog tipa podataka može mijenjati iz verzije u verziju. Tako u ovom slučaju gdje se Python aplikacija oslanja na paket za razvoj, postoji mogućnost da određene funkcije više neće biti dostupne nadogradnjom Python verzije ili u novijoj verziji vCenter servera.

S obzirom na obujam ovog rada, PowerCLI rješenje bilo bi kvalitetnije i učinkovitije izvedeno od Python-a, no želja autora je bila da se čitatelja upozna i sa drukčijim pristupom konfiguriranja VMware virtualne infrastrukture. Automatizaciju izrade i dohvaćanje korisnika iz Windows Active Directory-ja je putem PowerShell-a kudikamo lakše izvesti nego korištenjem dodatnih modula unutar Python okruženja. S obzirom da je PowerShell okvir za upravljanje kompletnom Windows infrastrukturom, spajanje na Windows Server i kreiranje korisnika moguće je izvesti i sa klijentskih računala. Kako određene funkcije u

trenutku pisanja ovog rada i dalje nisu dostupne putem API poziva, za izradu kompletnog rješenja potrebno je koristiti i druge programske jezike te ih ukomponirati unutar Python koda. Za razliku od PowerShell-a, takav pristup je bitno lakše primijeniti unutar Python-a jer korištenjem odgovarajućih modula, možemo pisati kod u drugom programskom jeziku te ga potom zapakirati u .py datoteku. Izrada korisnika i upravljanje dopuštenjima nije pokrivena VMware vSphere REST API dokumentacijom, stoga možemo zaključiti da trenutno nije ni dostupna putem tih kanala. Međutim, iste radnju bez problema možemo odraditi korištenjem PowerCLI modula u kombinaciji s PowerShell kodom.

Bez obzira na potencijalne nedostatke, Python se pokazao kao iznimno moćan alat za kreiranje aplikacije za skriptiranje izrade virtualnih mašina. Uz odgovarajuće module, konfiguracija virtualnih mašina postaje mačji kašalj, dok briga oko konfiguracije mreža i upravljanje pohranom postaje stvar prošlosti za krajnje korisnike. Pomoću aplikacije i dijete može kreirati funkcionalnu virtualnu mašinu, a pažljivim odabirom pozadinskih provjera možemo osigurati da je sve odrađeno prema potrebnim parametrima. S viskom nivoom automatizacije te širokom lepezom konfiguracijskih mogućnosti možemo uvelike ubrzati repetitivne zadatke koji bi zahtijevali sate pri ručnoj konfiguraciji vSphere-a. Samim time ovakva aplikacija za upravljanje privatnim oblakom može uštedjeti vrijeme i novac, ali i olakšati upravljanje VMware infrastrukturom.

Popis slika

Popis tablica

Popis kôdova

Literatura

[1] BRIAN DESMOND, JOE RICHARDS, ROBBIE ALLEN, ALISTAIR G. LOWE-NORRIS; *ACTIVE DIRECTORY: DESIGNING, DEPLOYING, AND RUNNING ACTIVE DIRECTORY FIFTH EDITION,* https://www.amazon.com/Active-Directory-Designing-Deploying-Running/dp/1449320023 , listopad 2022.

[2] NICK MARSHALL, MIKE BROWN, RYAN JOHNSON; MASTERING VMWARE VSPHERE 6.7, https://www.amazon.com/Mastering-VMware-vSphere-Nick-Marshall/dp/1119512948, listopad 2022.

[3] LUC DEKENS, JONATHAN MEDD, GLENN SIZEMORE, BRIAN GRAF, ANDREW SULLIVAN, MATT BOREN; *VMWARE VSPHERE POWERCLI REFERENCE: AUTOMATING VSPHERE ADMINISTRATION*, https://www.amazon.com/VMware-vSphere-PowerCLI-Reference-Administration-ebook/dp/B019WTGZ3A, listopad 2022.

[4] STEVE JIN; *VMWARE VI AND VSPHERE SDK: MANAGING THE VMWARE INFRASTRUCTURE AND VSPHERE*, https://www.amazon.com/VMware-VI-vSphere-SDK-Infrastructure/dp/0137153635, listopad 2022.

[5] PYTHON DOKUMENTACIJA, https://www.python.org/doc/, listopad 2022.

[6] VMWARE API DOKUMENTACIJA, https://developer.vmware.com/, listopad 2022.

[7] VSPHERE API DOKUMENTACIJA, https://developer.vmware.com/apis/vsphere-automation/latest/, listopad 2022.

[8] VMWARE PRIMJERI KODOVA, https://code.vmware.com/samples, listopad 2022.

[9] RED HAT BLOG, https://www.redhat.com/en/topics/virtualization/what-is-virtualization, listopad 2022.

[10] EDUCBA BLOG, https://www.educba.com/virtualization-platforms/, listopad 2022

[11] RED HAT BLOG, https://www.redhat.com/en/topics/cloud-computing/what-is-iaas, listopad 2022.

[12] VMWARE BLOG, https://www.vmware.com/topics/glossary/content/private-cloud-vs-public-cloud.html, listopad 2022.

[13] VMWARE BLOG, https://www.vmware.com/topics/glossary/content/bare-metal-hypervisor.html, listopad 2022.

[14] GITHUB PROJEKT, https://github.com/Magneet/Various_Scripts, listopad 2022.

[15] GEEKSFORGEEKS BLOG, https://www.geeksforgeeks.org/virtualization-cloud-computing-types/, listopad 2022.

[16] BMC BLOG, https://www.bmc.com/blogs/saas-vs-paas-vs-iaas-whats-the-difference-and-how-to-choose/, listopad 2022.

[17] GEEKSFORGEEKS BLOG, https://www.geeksforgeeks.org/pros-and-cons-of-virtualization-in-cloud-computing/, listopad 2022.

[18] BITPIPE BLOG, http://docs.media.bitpipe.com/io_24x/io_24693/item_407991/HP_sWindowsServer_SO%23033548_E-Guide_042011.pdf, listopad 2022.

[19] HEROIX BLOG, https://www.heroix.com/blog/vmware-vcpu-over-allocation/, listopad 2022.

[20] GARTNER BLOG, https://www.gartner.com/reviews/market/server-virtualization, listopad 2022.

[21] VMWARE BLOG, https://www.vmware.com/, listopad 2022.

[22] VMWARE DOKUMENTACIJA, https://docs.vmware.com/en/VMware-vSphere/6.5/vsphere-esxi-vcenter-server-65-resource-management-guide.pdf, listopad 2022.

[23] VMWARE DOKUMENTACIJA, https://docs.vmware.com/en/VMware-vSphere/7.0/vsphere-esxi-70-installation-setup-guide.pdf, listopad 2022.

[24] MICROSOFT DOKUMENTACIJA, https://learn.microsoft.com/en-us/windows-server/identity/ad-ds/get-started/virtual-dc/active-directory-domain-services-overview, listopad 2022.

[25] VMWARE DOKUMENTACIJA, https://docs.vmware.com/en/VMware-vSphere/7.0/com.vmware.vsphere.vm_admin.doc/GUID-CA948C69-7F58-4519-AEB1-739545EA94E5.html, listopad 2022.

[26] VMWARE DOKUMENTACIJA, https://docs.vmware.com/en/VMware-vSphere/7.0/com.vmware.vsphere.networking.doc/GUID-7225A28C-DAAB-4E90-AE8C-795A755FBE27.html, listopad 2022.

[27] GOVMOMI PROJEKT, https://github.com/vmware/govmomi, listopad 2022.

[28] VMware vSphere struktura, https://vdc-repo.vmware.com/vmwb-repository/dcr-public/6b586ed2-655c-49d9-9029-bc416323cb22/fa0b429a-a695-4c11-b7d2-2cbc284049dc/doc/vim.vm.Summary.html, listopad 2022.

[29] VCLOUDNINE BLOG, https://www.vcloudnine.de/first-steps-with-python-and-pyvmomi-vsphere-sdk-for-python/, listopad 2022.

[30] PYVMOMI PROJEKT, https://github.com/vmware/pyvmomi, listopad 2022.

[31] PYVMOMI PRIMJERI, http://vmware.github.io/pyvmomi-community-samples/, listopad 2022.