REPUBLIKA E SHQIPËRISË
UNIVERSITETI POLITEKNIK I TIRANËS
FAKULTETI I TEKNOLOGJISË SË INFORMACIONIT
DEPARTAMENTI I INXHINIERISË INFORMATIKE

ENKLI YLLI

PËR MARRJEN E GRADËS
"DOKTOR"
NË: "TEKNOLOGJITË E INFORMACIONIT DHE KOMUNIKIMIT"
DREJTIMI "INXHINIERI INFORMATIKE"

DISERTACION

SHKALLËZUESHMËRIA DHE EFIÇENCA E BAZAVE TË TË DHËNAVE NË CLOUD COMPUTING

Udhëheqës shkencor
Prof.Asoc. IGLI TAFA

Tiranë, 2021

SHKALLËZUESHMËRIA DHE EFIÇENCA E BAZAVE TË TË DHËNAVE NË CLOUD COMPUTING

Disertacioni
i paraqitur në Universitetin Politeknik të Tiranës
për marrjen e gradës
ñDoktorò
në:
ñTeknologjitë e Informacionit dhe Komunikimitò
Drejtimi ñInxhinieri Informatikeò

Nga

Z. Enkli Ylli
2021

JURIA PËR VLERËSIMIN E DISERTACIONIT PËR FITIMIN E GRADËS SHKENCORE ñDOKTORò

Miratuar
Me vendimin e Kyshillit tyProfesoryve tyFTI-syNr: 17, daty: 15.07.2021.

Juria:

Kryetar i jurisë:	Prof.Dr. Aleksandër XHUVANI
Anëtar i jurisë:	Prof. Dr. Elinda MEÇE
Anëtar i jurisë (Oponent):	Prof. Asoc. Indrit ENESI
Anëtar i jurisë:	Prof. Asoc. Marenglen BIBA
Anëtar i jurisë (Oponent):	Prof.Dr. Luan KARÇANAJ

Dekan i Fakultetit tyTeknologjisysyInformacionit

Prof. Dr. Elinda MEÇE

Falënderime

Ky punim nuk do të ishte i mundur pa bashkëpunimin dhe orientimin e paçmuar të udhëheqësve të mi Z. Igli Tafa dhe Z. Neki Frashëri. Eksperienca dhe këshillat e tyre kanë qenë një gur themeli në atë cka kam bërë dhe projektuar unë gjatë kitj punimi.

Falenderoj stafin e Departamentit të Inxhinierisë Informatike për sugjerimet dhe orientimet e tyre gjatë prezantimeve të mbajtuar në Departament.

Falenderoj prindërit e mi për nxitjen për të ndermarrë këtë aktivitet të rëndësishëm në jetën time profesionale si dhe bashkëshorten time, Blerinën për mbështetjen, durimin dhe këshillat e sajrajtën përfundimtare këtij punimi.

Përmbajtja

LISTA E TABELAVE

LISTA E FIGURAVE

Abstrakt (Shqip)

Ky punim ka për qëllim analizimin e shkallëzueshmërisë dhe lasticitetin e aplikimeve dhe bazave të të dhënave në Cloud. Me anë të këtij punimi do të realizojmë një analizë në raport me menaxhimin e shkallëzueshmërisë në Cloud pa afektuar performancën duke mundësuar shpenzime sa më të vogla për arkitekturat ku pagesa bëhet sipas përdorimit.. Punimi yne do të fokusohet drejt një menaxhimi sa më efektiv në performancë duke synuar një menaxhim sa më të mirë të kostove gjithashtu.

Në dy dekadat e fundit ka patur një rritje eksponenciale të përdorimit të internetit në çdo skaj të globit. Shtimi i rrjeteve sociale, mediave interaktive, transmetimeve në kohë reale i kanë bërë njerëzit edhe më të varur nga interneti por edhe me eficientë njëkohësisht në realizimin e punëve, edukimit, argëtimit etj. Epoka digjitale tashmë është një realitet dhe aksesimi i internetit në çdo skaj të botës dhe nga grup mosha të ndryshme e bën shpeshherë trafikun të paparashikueshëm. Interesat e çdo përdoruesi janë nga më të ndryshmet. Interesa personale apo trafik që konvergjon drejt një dstinacioni të vetëm mund të shkaktojë një mbingarkesë në aplikim. Është një sfidë menaxhimi i burimeve të informacionit në mënyrë që ato të jenë të disponueshme dhe të ofrojnë një performancë të pritur. Ky proces modernizimi dhe qëndrueshmërie është një nga objektivat kryesorë të të gjithë ofruesëve të aplikimeve. Shumë modele apo udhëzues për ngritjen e aplikimeve që ofrojnë disponueshmëri egzistojnë. Këto modele janë të aplikueshme si në rastin e serverëve fizikë dhe të virtualizuar. Për ofrimin e një aksesi më të shpejtë drejt aplikimeve është e gjithë pranuar që kjo mundësohet vetëm në Cloud.

Cloud computing është një platformë që ofron fleksibilitet, disponueshmëri, modernizim dhe komoditet në menaxhim të shërbimeve si për bizneset e vogla dhe të mesme ashtu dhe ato të mëdha. Cloud computing ofron mundësinë e ngritjes së së aplikimeve me disponueshmëri të lartë në mënyre më të thjeshtë se në menaxhimin klasik të infrastrukturave. Këto lloj aplikimesh mund të përballojnë disa milion përdorues në kohë reale p.sh ebay.com, amazon.com, aliexpress.com. Ky punim ka si qëllim analizën, hulumtimin dhe zhvillimin e metodave që mundësojnë shkallëzueshmëri të lartë për

aplikimet dhe bazat e të dhënave në Cloud. Ky punim analizon shkallëzueshmërinë e aplikimeve si në rritje ashtu dhe në zbritjeve në mënyrë të automatizuar duke analizuar dhe kostot mbi të gjitha këto veprime me qëllimin e vetëm për mbajtjen e performancës në një nivel të kënaqshëm dhe kontrollin ndaj kostove në mënyrë efektive.

Abstrakt (Anglisht)

In the last two decades there has been an exponential increase in internet usage in every corner of the globe. The addition of social networks, interactive media, real-time broadcasts have made people even more dependent on the Internet but also more efficient at the same time in the realization of work, education, entertainment, etc. The digital age is already a reality and internet access in every corner of the world and from different age groups often makes traffic unpredictable. The interests of each user are very different. Personal interest or traffic converging towards a single destination may cause an overload on the application. It is a challenge to manage information resources so that they are available and deliver the expected performance. This process of modernization and sustainability is a challenge for all application providers. There are many models or guides for setting up applications that provide availability. These models are applicable to both physical and virtualized servers. For faster access to applications it is generally accepted that this is only possible in the Cloud.

Cloud computing is a platform that offers flexibility, availability, modernization and convenience in managing services for both small and large businesses. Cloud computing offers the ability to set up high-availability applications in a simpler way than in classic infrastructure management. These types of applications can accommodate several million users in real time eg ebay.com, amazon.com, aliexpress.com. This paper aims to analyze, develop a nd investigate algorithms and mrthods to implement high scalability applications in Cloud environments. This paper analyzes the scalability of both increasing and decreasing applications automatically by analyzing the costs of all these actions with the sole purpose of keeping performance at a satisfactory level and controlling costs effectively.

SHKURTIME

Cloud Computing: Cloud

Marrëveshja për Nivelin e Shërbimeve (Service Level Agreement: SLA)

Niveli i Shërbimit të Operimit (Service Level Operation: SLO)

U.S National Institute of Standard and Technology: NIST

Sistemi Operative (Operation System :OS): SO

Random Access Memory: RAM

Wide Area Network: WAN

Infrastructure as a Service: IaaS

Platform as a Service: PaaS

Software as a Service: SaaS

Balancimi I Ngarkesës (Load Balancer: LB)

Virtual Private Network: VPN

Central Processing Unit: CPU

Menaxheri I Makinës Virtuale (Virtual Machine Monitor: VMM)

Amazon Web Services: AWS

Makina Virtual (Virtual Machine: VM): MV

Cilësia e Shërbimit (Quality of Service: QoS)

Ofruar: O

Kërkuar: K

Shkallëzim Vertikal (Vertical Scaling: VS)

Shkallëzim Horizontal (Horizontal Scaling: HS)

Modeli I Sistemit (System Model: SM): MS

Virtual Machine Live Migration: VMLM

Migrimi i Drejtpërdrejte i Aplikacionit (Automatic Live Migration: ALM)

Kthimi i Investimit (Return on Investment: ROI):

Monitoring Analysis Planning Execution: MAPE

Treguesit Kyc të Performances (Key Performance Indicator: KPI)

Information Technology: IT

Amazon Web Services: AWS

Elastic Cloud Compute: EC2

Input/ Output: I/O

Application Programming Interface: API

Application Server: AS

Database Server: DS

Database: DB

Internet Protocol: IP

Node Controller: NC

Cluster Controller: CC

Cloud Controller: CLC

KAPITULLI I

Informacion i përgjithshëm dhe organizimi i punimit

Hyrje

Në këtë kapitull do të jepet informacion i përgjithshëm por i nevojshëm nga ana konceptuale për të kuptuar edhe më shumë fokusin e temës. Në këtë kapitull do të jepet një informacion mbi motivimin si dhe pyetjet që kemi ngritur dhe që do të trajtoj në këtë punim doktoral.

1.1. Motivimi

Në vitin që u angazhova për fillim e doktoratës trendi kryesor në fushën e menaxhimit të sistemeve kompjuterike ishte virtualizimi dhe ñCloud Computingò. Virtualizimi dhe ñCloud Computingò shihej si një trend për shfrytëzimin maksimal të serverëve fizikë për të vendosur dy apo më shumë sisteme në një server. Në treg egzistonin kompani si Amazon që kishin filluar të ofronin shërbime hostimi serverësh për palë të treta. Ky lloj shërbimi lindi si ide nga themeluesit e kompanisë Amazon për të krijur një platformë të përbashkët teknologjike për keipet e saj inxhinjerike[93]. Duke patur shumë aplikime në përdorim, ata kishin një ekip inxhinjerik për menaxhimin e gjithë shërbimeve harduare dhe software brenda saj. Duke parë nevojën për zhvillim në treg ata vendosën që këtë paltformë të ngritur për interesat e tyre ta bënin publike dhe për zhvilluesit e tjerë me parimin ñe shpejtë, e lirë dhe e besueshmeò. Kjo është me pak fjalë fillesa e Cloud komercial. Shtimi i përdorueseve të këtyre burimeve orientoi ofruesit e Cloud të modernizojnë mënyrën e menaxhimit të Cloud duke i dhënë cilësi shtesë. Siguria dhe menaxhimi i aplikimeve në Cloud, një

ambient jashtë dhomës së serverëve, koncepti klasik i një administratori IT ku cdo gjë është nën kontroll dhe e prekshme u bë dhe një nga motivet e mia kryesore për transferimin e zgjidhjeve lokale në ato Cloud. Ndërgjegjësimi gjithashtu mbi kostot e blerjeve, mirëmbajtjes dhe shpenzimit të energjisë u bënë një nga arësyet më të qenësishme që shume biznese i kaluan shërbimet e tyre në Cloud me qëllim uljen e shpenzimeve kapitale dhe transferimin vetëm të një pjese buxheti të tyre në kostot operative. Shkallëzueshmëria është një veti që ndihmon në procesin e uljes së kostove në rast se përdoren në mënyrën e duhur.

Të gjitha këto veçori që ishin një koncept i ri për të gjithë ne si profesionistë në këto vite shoqëruar dhe me një nga parimet kryesore të IT, që është disponueshmëria e sistemve më bënë që të fokusohesha në këtë temë me dëshirën për të marrë informacionin maksimal në lidhje me këtë teknologji, shfrytëzimin optimal të vetive të kësaj teknologjie kundrejt disponueshmërisë së sistemeve në ñCloud Computingò, dhënien e një kontributi në ofrimin e një optimizimi drejt në menaxhimi të rregulluar ku bashkimi i dy qëllimeve, shakllëzueshmërisë dhe menaxhimi i kostove të kthehen në një kornizë që u jep mundësinë të gjithëve të përfitojnë nga kjo.

1.2. Qëllimi dhe kontributet e këtij punimi

Qëllimi i kësaj teme është shfrytëzimi maksimal i vetive dhe opsioneve të ofruara nga Cloud për ngritjen e një algoritmi automatik për menaxhimin e shkallëzueshmërisë së sistemeve (aplikime, baza të dhënash etj) në Cloud me qëllim mbajtjen e kostove operacionale në nivelin më të ulët duke ofruar cilësi shërbimi të kënaqëshme pavarësisht kërkesave ne sistem. Në këtë punim do të ofrojmë një këndvështrim të administrimit të shkallëzueshmërisë në Cloud si dhe do të ngremë një kornizë llogjike si dhe një implementim në një Cloud privat të ngritur nga unë për matjen e efektivitetit të këtij algoritmit.

Metodologjia e aplikuar në këtë punim është:

- Studimi i literaturës aktuale në lidhje me menaxhimin e shkallëzueshmërisë në Cloud. Nëpërmjet abonimeve në revista shkencore si dhe ne faqe publikimesh shkencore të fushës, kam hulumtuar literaturën e nevojshme mbi fushën dhe objektin e kësaj teme. Kjo literaturë dhe publikimet shkencore më kanë shërbyer për të ngritur një bazë të qëndrueshme teorike e cila më ka shërbyer në rastet e ngritjes së hipotezës dhe pyetjes kërkimore mbi të cilat do të jetë edhe kontributi i këtij punimi.
- Studimi i opsioneve të servirura nga ofruesit e Cloud për menaxhimin e shkallëzueshmërisë në Cloud. Gjatë kësaj periudhe në ambientin ku unë zhvilloj aktivitetin e përditshëm janë konceptuar dhe analizuar zgjidhje teknologjike mbi ambientin Cloud. Personlaisht në eksperiencën time jam përpjekur të bëj lidhjen teorike me eksperiencën praktike për të parë më në detaj vecoritë dhe të përbashkëtat të ofruesëve të shërbimeve Cloud në botë.
- Vlerësimi i opsioneve të skallëzueshmërisë për përdoruesit e Cloud që është mundësuar duke u futur si një përdorues i Cloud publik dhe analizimin e opsioneve të shkallëzueshmërisë me avantazhet dhe disavantazhet që jepen, e parë në fokusin e biznesit dhe lehtësinë e operimeve në Cloud.
- Vlerësimi i parametrave monitoruese në Cloud si burime të vlefshme në vendimmarrje. Ky vlerësim është bërë në kudarin praktik por dhe duke marrë për bazë vlerësime të pavarura nga analistë të shërbimeve Cloud,
- Shpejtësia e reagimit dhe varësia e trye nga monitorimi si proces. Kjo është dhe një nga pjesët më të rëndësishme ku fokusohet dhe punimi.

Kontributet e këtij punim janë:

- Krijimi i një algoritmi që realizon shkallëzueshmëri vetëm në shtresat ku ka rënie performance. Ky algoritëm përshtatet me kërkesat e paraqitura dhe modelimin e serverëve sipas kapacitetit maksimal të kërkesave që ata mund të përballojnë në raport me parametra e vetë serverit. Profilizimi do të na shërbejë dhe për të menaxhuar elasticitetin duke parë numrin e kërkesave në hyrje që do të raportohen nga LB nëpërmjet sistemit të monitorimit.

- Menaxhim i shkallëzueshmërisë me orientim drejt kostove. Në disertacion sic do të shihet edhe më qartë gjatë aplikimit dhe marrjes së rezultateve përfundimtare menaxhimi i shkallëzueshmërisë do të bëjë një përqasje të orientuar ndaj kostove kur të gjendet para veprimit të ñScale-Upò ose ñScale-Downò. Efekti i algoritmit dhe përzgjedhja e tipit të instancës në mënyrë të tillë që të plotësohen kërkesat në hyrje me raportin e kërkesave të përballueshme nga sistemi. Përzgjedhja e instancës me parametrat më të vegjël për të kënaqur kërkesat në hyrje do të japë efekt në tarifim dhe rrjedhimish do të kemi kosot më të ulëta në egezekutimin e aplikimit.
- Menaxhim automatik i shkallëzueshmërisë. Menaxhimi automatik i shkallëzueshmërisë që ndryshe përkufizohet elasticiteti nëpërmjet punimeve që jemi bazuar por duke e avancuar modelin me qëllim përzgjedhjen e makinave virtuale sipas profilizimit me mjete alterantive nga ato të përdorura në algoritmin që jemi bazuar.
- Vlerësimi i mjeteve të monitorimit të Cloud. Përparësia e sistemeve të integruar në Cloud me ato që përdoren si mjet i përgjithshëm monitorimi. Një konkluzion si dhe një pohim për rëndësinë e përdorimit të mjeteve të integruara në Cloud.

1.3 Organizimi i punimit

Ky disertacion është i organizuar si më poshtë:

Në Kapitullin e I jepet informacioni i përgjithshëm mbi temën, punimin, motivimin dhe kontributet e këtij punimi.

Në Kapitullin e II jepet informacion i përgjithshëm mbi virtualizimin, Cloud, mënyrat e organizimit. Ky kapitull përbën bazën për krijimin e koncept idesë për të vijuar më tej me thellimin e studimit. Ai është i rëndësishëm për kuptimin e sfidave në menaxhimin e shkallëzueshmërisë dhe elasticitetit.

Në Kapitullin e III do të flasim gjerësisht për elasticitetin dhe shkallëzueshmërinë në ambientet Cloud. Do të falsim për avantazhet, mënyrat e aplikimit të teknikave të elasticitetit dhe shkallëzueshmërisë në Cloud.

Në Kapitullin e IV do të jap bazën e literaturës se ku jam bazuar në punimin tim dhe që më ka shërbyer më së shumti në zgjerimin e njohurive të mia teorike dhe thellimin e kërkimit në drejtimin e shkallëzueshmërisë në Cloud.

Në Kapitullin e V do të fokusohemi tek algoritmi i propozuar si dhe do të bëjmë shpjegimin mbi hapat dhe vendimmarrjen që realizon algoritmi në bazë të monitorimit të sistemeve që përbëjnë aplikacionin, serverëve dhe LoadBalancers.

Në Kapitullin e VI do të japim rezultatet e eksperimentit si dhe krahasimin në lidhje me algoritmin dhe implementimin Kingfisher si dhe mejtet e profilizimimit të serverëve TCP-W si dhe Quelea.

Në kapitullin e VII do të japim konkluzionet dhe mundësitë e zhvillimit në të ardhmen që mund të bëhen duke u bazuar në teorinë e këtij punimi.

KAPITULLI II

Cloud Computing, organizimi, modelet

2.1. Hyrje

Në këtë kapitull do të japim një informacion bazë mbi virtualizimin, ñCloud Computingò si dhe një përcaktim mbi aplikimet me shumë shtresa. Në lidhje me ñCloud Computingò do të shpjegojmë grupimet dhe modelet kryesore të ñCloud Computingò.

2.2. Bazat e virtualizimit dhe Cloud Computing

Tradicionalisht ka qenë shumë e shtrenjtë, e komplikuar dhe e vështirë për bizneset të mbanin aplikacione të ndryshme sepse ekzekutimi i tyre kërkonte një ekip të tërë ekspertesh për të instaluar, konfiguruar, testuar, ekzekutuar, siguruar dhe azhornuar këto aplikime dhe sistemet ku janë të vendosura ato.

Pikërisht ñCloud Computingò i siguron kompanive ose individëve infrastrukturën, softuerin dhe platformat e nevojshme për të vendosur atje aplikimet dhe/ose sistemet e tyre duke iu mundësuar këtyre kompanive të fokusohen në shërbimet që ato ofrojnë vetë dhe duke iu hequr shqetësimin për blerjen, licensimin, mirëmbajtjen dhe zgjidhjen e problemeve harduarike.

Shërbimet Cloud po behën gjithnjë e më prezentë në ditët tona. Por ky term ka ekzistuar shumë më parë seç ne e hasim ne ditët e sotme në koncept apo implementim. Për herë të parë ky term është hasur në dokumentet e brendshme të Compaq [94]. Për sa i përket ñCloud Computingò, në nivel fizik ai mbështetet në të njëjtën shtresë fizike (servera). Abstragimi fillon në shtresat dhe sistemet e implementuara mbi pjesën fizike. Në ditët e

sotme të gjithë marrim shërbime nga këto sisteme, të implementuara nga ofruesit e shërbimeve.

2.2.1 Çfarë është Cloud computing?

Kompanitë e ndryshme kanë përkufizimin e tyre për Cloud dhe Cloud computing, por shumica e këtyre përkufizimeve përqendrohen tek disa atribute të rëndësishme; siç janë burimet e kërkuara sigurohen shpejt, sipas kërkesës, shërbimi është i shkallëzuar për një përdorim optimal të burimeve dhe konsumatori paguan vetëm për atë që përdor. Pra sipas një studimi, Cloud është një grup i madh burimesh të virtualizuara lehtësisht të përdorshme dhe të arritshme (të tilla si hardueri, platformat e zhvillimit dhe / ose shërbimet e tjera). Këto burime mund të rikonfigurohen për t'u përshtatur me një ngarkesë të ndryshueshme, duke lejuar gjithashtu përdorimin optimal të burimeve [1]. Ky grup burimesh zakonisht shfrytëzohet nga një model përdoruesi me pagese e cila garantohet nga Ofruesi i Infrastrukturës me anë të Marrëveshjes së Nivelit të Shërbimit (Service Level Agreement:SLA) të personalizuara [1].

Instituti Kombëtar i Standardeve dhe Teknologjisë së Shteteve të Bashkuara të Amerikës (U.S. National Institute of Standard and Techonogy ï NIST) e përkufizon Cloud computing si modeli që krijon mundësinë për të aksesuar një grup të përbashkët burimesh kompjuterike të konfiguruara (si p.sh. rrjetet, serverat, hapësira ruajtjeje, aplikacionet dhe shërbimet) kurdoherë, sipas nevojave dhe në bazë të kërkesës dhe që mund të sigurohen dhe mund të hiqen me shpejtësi me përpjekje minimale menaxheriale [27].

Gjithsesi, njohuri më të thelluara mbi Cloud Computering dhe karakteristikat e tij do të jepen në seksionet e tjera të këtij punimi. Seksioni vijues do të japë informacion mbi modelet dhe shërbimet e tij. Një nga pjesët kryesore të Cloud Computing është virtualizimi. Virtualizimi është një metodë qe po përdoret gjerësisht ne sistemet Cloud. Kjo teknike përdoret për të ndarë burimet e ndryshme si për shembull sistemin operativ, memorien, serverin apo rrjetin, duke përmirësuar kështu përdorimin e tyre në nivel maksimal, gjithashtu duke ulur kostot ne infrastrukturën IT dhe duke rritur abstraksionin. Më shumë detaje rreth virtualizimit do të jepet në seksionin përkatës.

Shpjegimi i dhënë më sipër për Cloud Computing nënvizon disa aspekte të rëndësishme:

Së pari, Cloud Computing nuk është një teknologji e re por në të vërtetë është një model i ri i aksesimit dhe përdorimit të burimeve me shkallezueshmeri të lartë;

Së dyti, Cloud Computing është bërë e mundur me zhvillimin e teknologjisë si p.sh virtualizimi, alokimi dhe konfigurimi i burimeve në mënyre dinamike, fuqia e procesimit (CPU, RAM, kapaciteti), aksesimi ne mënyrë të shpejtë dhe të qëndryeshëm në rrjet (LAN dhe WAN) dhe së treti, shërbimet kompjuterike përdoren dhe ofrohen si shërbime nëpërmjet internetit.

Por si mund ti percaktojmë Cloud Computing dhe llojet e infrastrukturave që ekzistojnë. Cloud Computing po ofron gjithnjë e më shumë shërbime shtesë me të cilat pasurohet dhe përmirësohet niveli i shërbimeve qe një ofrues mund tëju japë klientëve të tij. Këto jo vetëm në nivel shërbimi por edhe sigurie, qëndrueshmërie ku gjithashtu jepet mundësia e vizibilitetit edhe më të madh gjate monitorimit të shërbimeve të marra nën përdorim.

2.2.2 Karakteristika e Cloud Computing

Shumë studime[1],[2] kanë analizuar karakteristikat e Cloud Computing dhe çfarë e dallon Cloud Computing nga format tradicionale. Përkufizimi i Institute Kombëtar I Standardeve dhe Teknologjisë së Shteteve të Bashkuara të Amerikës (U.S. National Institute of Standard and Techonogy ï NIST) [8] përshkruan karakteristikat e mëposhtëme të modelit Cloud Computing.

Vetë-shërbim sipas kërkesës (On- demand self-service): Përdoruesit mund të sigurojnë automatikisht burimet e tij kompjuterike sipas nevojës. Kjo arrihet pa ndërhyrja njerëzore, zakonisht përmes një portali ndërveprues që u mundëson përdoruesit të konfigurojnë dhe menaxhojnë vetë këto shërbime [4]. Pra, përdoruesit mund të marrin në mënyrë automatike shërbimet dhe për të arritur këtë kërkohet që (1) Shërbimi duhet të jetë gjithnjë i disponueshëm ; dhe që (2) Shërbimi i marrë duhet të modifikohet nga klienti pa kontaktuar me ofruesin e tij [5]. Pra në këtë rast përdoruesi nuk është i limituar vetëm tek

një server i specifikuar. Kjo karakteristikë është një cilësi thelbësore sepse përdoruesit ose kompanitë nuk e kanë të domosdoshme të planifikojnë rritjen ose zvogëlimin e tyre për karakteristikat e tyre teknike.

Aksesi i gjerë në rrjet (Broad network access): Aksesi i gjerë në rrjet do të thotë se burimet duhet të jetë e aksesueshëm ose disponueshëm nga çdo pajisje rrjeti [5]. Pra nuk ka rëndësi se çfarë pajisje përdor për të aksesuar burimet qoftë një telefon inteligjentë, laptop, tablet apo desktop dhe çfarëdolloj linje interneti, ofruesi duhet të ofrojë internet dhe aksuesshmëri në cdo moment.

Elasticiteti i Shpejtë (Rapid Elasticity): Burimet mund të zgjerohen me shpejtësi dhe në mënyrë transparente ose mund të merren në varësi të kërkesës. Pra, Cloud shfaqet pa limite për përdoruesit dhe ata mund të marrin me qira ose të çlirojnë burimet e Cloud sipas nevojës së tyre. Shkallëzimi është automatik për përdoruesit dhe ofrimi i asaj që kanë nevojë përdoruesit e Cloud është transparente. Shkallëzimi i burimeve mund të bëhet në disa mënyra si p.sh. shkallëzimi vertikal ose horizontal të cilat do të shpjegohen në kapitullin tjetër.

Monitorimi i Shërbimit (Measured Service): Pra duke qenë se Could Computing ofron vetë-shërbim sipas kërkesës, përdorimi i saj kontrollohet dhe monitorohet në mënyrë automatike duke përdorur aftësitë matëse nga ofruesit e shërbimeve Cloud. Shërbimet matëse përfshijnë disa lloje burimesh Cloud sic janë: kapacitetin, fuqia procesuese, gjerësia e brezit dhe llogaritë aktive. Shërbimet matëse janë esenciale për faturimin e saktë të burimeve të përdorura.

Grumbullimi i burimeve (Resource pooling): Burimet dhe shërbimet kompjuterike të një Ofruesi të Shërbimeve Cloud mund të grupohen për ti shërbyer disa përdoruesve duke përdorur modelin multi-qiramarrës dhe teknologjinë virtualizuese. Pra, burimet e ndryshme virtuale dhe fizike mund të alokohen dhe mund të hiqen sipas kërkesës së përdoruesit. Përdoruesi i Cloud nuk ka njohuri ose kontroll mbi vendndodhjen e burimeve. Gjithashtu, kjo lejon aksesimin e burimeve të mëdha si për përdoruesit edhe kompanitë sesa prokurimi i drejtpërdrejte vetë i infrastrukturës fizike ose virtuale. Sipas një studimi, principi i grumbullimit të burimeve lejon së pari grupimin e tyre në mënyrë

praktike me një kosto të pranueshme dhe së dyti grupimin e burimeve me kosto efektive duke pasur fleksibilitet dhe përdorim të lartë të burimeve[6].

2.2 Shtresat e Cloud Computing

Sipas Cloud Security Alliance [69], Cloud mund te modelohet në 7 shtresa [7] të cilat mund të kontrollohen nga një përdorues ose ofruesi i tij:

1. Ambienti - në këtë shtresë kemi parasysh infrastrukturën fizike që përbëhet nga qendrat e të dhënave ku qëndrojnë pajisjet e rrjetit dhe kompjuterit
2. Rrjeti - këtu kemi lidhjet e rrjetit si brenda përbrenda Cloud fizikisht dhe virtualisht ashtu edhe lidhja midis Cloud dhe përdoruesit.
3. Harduare - këtu kemi parasysh komponentët fizikë të pajisjeve të rrjetit dhe kompjuterëve
4. Sistemi operativ (Operation System: OS) - në këtë shtresë ne konsiderojmë përbërësit e softuerit që formojnë sistemin operativ si për supervizorin ashtu edhe për makinat
5. Shtresa e ndërmjetme (middleware) ï është shtresa e softuerit midis Sistemit Operativ dhe aplikacionit të përdoruesit, zakonisht kjo shtresë është e pranishme vetëm në modelet SaaS dhe PaaS.
6. Aplikimi - këtu kemi aplikimin e ekzekutuar nga përdoruesit e sistemit Cloud.
7. Përdoruesi - në këtë shtresë ne konsiderojmë përdoruesit e sistemit Cloud dhe aplikacionet që ekzekutohen jashtë Cloud (p.sh. një shfletues interneti që ekzekutohet në premisën e përdoruesit).

Modelet e Shërbimit Cloud (Cloud Service Model)

Infrastrukturat e Cloud Computing mund ti ndajmë në mënyrën e implementimit dhe aksesit si dhe në tipin e shërbimit ose çfarë ofrojnë ato në vetvete. Ofruesit e Cloud ofrojnë shërbime Cloud duke i dhënë përdoruesve të tyre pak a shumë kontroll mbi burimet e ofruara në varësi të llojit të shërbimit cloud. Kur përdoruesi zgjedh një Ofrues Cloud, ato

mund të krahasojnë shërbimet e mundshme të ofruara çdo ofruesi Cloud me nevojat e tyre. Përdoruesit duhet të jenë të vetëdijshëm që ofruesi i tyre i Cloud do ti tarifojë ato mbi bazën e përdorimit. Kjo do të thotë që përdoruesi duhet të marrë me qira shërbime të tjera ose mund të shkëpusë disa shërbime në varësi të nevojave të tij ne çdo kohë. Më poshtë do të përshkruhen tre llojet e modeleve Cloud [8] dhe aktorët e përfshirë në to. Përdoruesi mund të zgjedhë midis: ñInfrastructure as a Serviceò (IaaS), ñPlatform as a Serviceò (PaaS) ose ñSofware as a Serviceò (SaaS). Këto tre lloj modelesh shërbimi ofrohen nga ofrues të ndryshëm Cloud. Siç u përmend edhe me lart, këto tre modele ndryshojnë nga kontrolli që përdoruesi ka në krahasim me Ofruesin e Cloud-it.

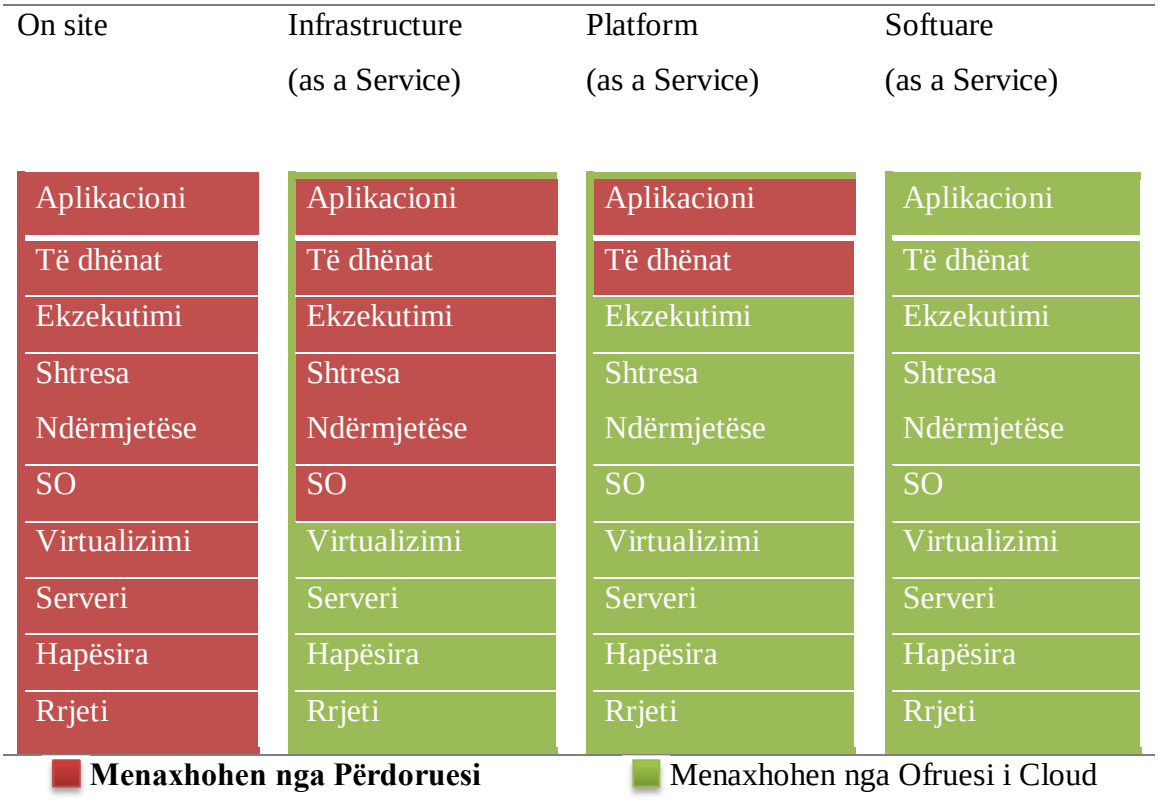


Figura 2 1:Modelet e Përdorimit të Cloud [9]

a. Aktorët

Ofruesit e Shërbimeve i bëjnë shërbimet të arritshme për përdoruesit e këtyre shërbimeve përmes faqeve dhe aplikimeve të vëna në dispozicion për atë qëllim. Cloud ka si qëllim të transferojë sigurinë e infrastrukturës informatike të kërkuar

tek një palë tjetër. Kjo infrastrukturë ofrohet si një shërbim nga ñOfruesi i Infrastrukturësò duke kaluar burimet kompjuterike nga ñOfruesi i Shërbimit/Infrastrukturësò tek përdoruesi.

b. Infrastructure as a Service (IaaS)

Në vetvete kjo është mënyra me e thjeshtë e ofrimit të një shërbimi Cloud. Nëpërmjet një virtualizimi serverash dhe shërbimesh, duke iu ofruar përdoruesve një ndërfaqe sa më të thjeshtë për menaxhimin e sistemeve te tyre, por njëkohësisht duke iu ofruar siguri në nivelin e aksesit dhe menaxhimit. Siç tregohet edhe në figurën 2.1, IaaS ndihmon përdoruesit duke u kujdesur për disa komponentë nga Virtualizimi deri tek rrjeti ndërsa përdoruesi është përgjegjës pjesën nga aplikacioni deri në sistemin operativ (SO).

IaaS siguron makinat virtuale ku aplikacionet dhe ndryshime mbi to mund të ekzekutohen. Ashtu sikurse ofruesit e Cloud janë përgjegjës për menaxhimin e infrastrukturës themelore të Cloud Computing, përdoruesit mund të shmangin kostot e kapitalit njerëzor, harduare, energjitikë etj. Ofrues kryesor i IaaS është Amazon Elastic Compute Cloud (EC2)[10]. Gjithashtu, ka edhe disa sisteme me burime te hapura kodi që janë zhvilluar, ku me cilat mund te virtualizojnë burimet fizike të IaaS Cloud siç është Nimbus[11], OpenNebula [12], Eucalyptus [13] etj.

Avantazhet:	**Disavantazhet:**
• shkallëzimi dinamik i infrastrukturës • koha e punës e garantuar • Automatizim i detyrave administrative • Balancimi i ngarkesës elastike(Elastic load balancing ELB) • Shërbime te bazuara ne politika	• Siguria softuare përbën rrezik te madh (Ofruesit e palëve te treta janë me te prirur për sulme) • Probleme me performancën si dhe shpejtësi e ngadaltë e lidhjes/aksesit

• Aksesim global	

c. Platform as a Service (PAAS)

Është një formë tjetër e përdorimit të burimeve cloud e cila prezantohet më shumë si një platformë software se sa një infrastrukturë e virtualizuar [14]. E vendosur një shtresë më lart se IaaS ky model ofron mundesinë e ngritjes së aplikimeve të përshtatura sipas nevojave. Sipas figurës, përdoruesit nuk kanë nevojë të blejnë dhe të menaxhojnë softuare dhe infrastrukturën nën të, por kanë autoritet mbi aplikacionet e vendosura dhe ndoshta konfigurimin e mjedisit te aplikacioneve. Përgjithsisht përdoruesit e PaaS janë zhvillues aplikacionesh të cilët i ofrojnë aplikacionet e tyre përdoruesëve të tjerë. Shëmbuj të ofruesve të PaaS jane Google App Engine[15], Microsoft Azure [16] etj.

Avantazhet:	**Disavantazhet:**
Å Vendosja e thjeshtuar e elementëve Å Funksionalitete të parandërtuara Å Rrezik i ulët Å Modeli Pay-per-use Å Shkallëzueshmëria	Å Kërkimi i një shitësi te ri kushton shumë prandaj nuk mund ta ndërrosh (Vendor lock-in) Å Privatësia e të dhënave Å Integrimi me pjesën tjetër të aplikacioneve te sistemit

d. Softuerë as a Service (SAAS)

Ky shërbim i Cloud Computing i ofron përdoruesve aplikacione softuerë sipas kërkesave nëpërmjet internetit dhe nuk kërkon instalimin në një kompjuter ose në një dhomë serverësh. Pra, SaaS ofron mundësinë për përdoruesit të përdorimit të një produkti final të ofruar nga prodhuesi. Sipas figurës, kjo lloj platforme çliron përdoruesin dhe ofruesin nga përdorimi i programit në infrastrukturën e klientit dhe ofruesi i Cloud merr përgjegjësinë nga pjesa e rrjetit deri tek aplikacioni. Kjo mundëson një menaxhim më të mirë për sa i përket sigurisë, licencimit si dhe ul kostot e platformave harduerë të dedikuar nga përdoruesi ku kostot operacionale menaxhohen nga ofruesi i platformës. SaaS është një model që i ofron përdorusit pavarësi nga paisja fundore[17].

Ofruesit tarifojnë mbi një bazë pagese për përdorim, nëpërmjet abonimit, marketingut ose duke e ndarë midis shumë përdoruesve. Aplikacionet SaaS me të përdorura janë Google Docs ose Calendar, Office 365, Salesforce CRM, Freshbooks, Basecamp, etj.)[18]

Avantazhet:	**Disavantazhet:**
• Çmim i ulet • Administrim më i lehtë • Aksesim global • Kompatibilitet (nuk kërkohet ndonjë harduerë apo softuerë i specializuar)	• Probleme me sigurinë dhe vonesën e transferimit te të dhenave • Varësi totale nga interneti • Kalimi nga një shitës SaaS në një tjetër është e vështirë

2.3 Modelet e Implementimit/ vendosjes (Deployment Models)

Pikërisht kjo zhvendosje e shpejtë e shërbimeve në Cloud Computing ka rritur kërkesën për të pasur modele të ndryshme se si Cloud implementohet. Modeli i vendosjes ose implementimit në Cloud i klasifikojmë në 4 tipe: 1) Publik, 2) Privat, 3) Hybrid Could dhe (4) Could Community të cilat do të paraqiten shkurtimisht më poshtë. Përzgjedhja e këtyre modeleve varet nga siguria e të dhënave të klientit dhe kërkesat par menaxhim.

a. **Public Cloud.** Public Could janë entitete të krijuara nga subjekte specifike për ofrimin e shërbimeve te caktuara kundrejt pagesës. Ky tip modeli është i dedikuar një organizate të caktuar ose grupi të caktuar. Sipas Kalpana Parsi dhe M.Laharika, Public Cloud është zgjedhja më e lëhtë për të konfiguruar dhe mirëmbajtur shërbimeve nga kompanitë e vogla [19]. Këto kompani shpesh nuk kanë shumë kapital dhe kanë më pak rrezik në humbjen e informacionit për shkak të vjedhjes ose shkeljes së sigurisë.p.sh Amazon, Azure, DigitalOcean, IBM Cloud, Oracle

b. **Private Cloud**. Janë infrastruktura që ngrihen vetëm për qëllime të brendëshme të kompanisë me qëllim vendosjen e shërbimeve në to. Si rrjedhojë, Private cloud është e dedikuar për një organizatë apo një grup të vetëm. Nuk ndahet me organizata të tjera. Private Could mund të jetë në pronësi ose është marrë me qira. Ky model është më i shtrenjtë dhe më i sigurt kur krahasohet me publikun. Pra, sipas përkufizimit të Armbrust-at-al, Private Cloud është një qendër të dhënash e brëndshme për një biznes ose organizatë e cila nuk është aksesueshme për publikun [20]. Private cloud kemi Eucalyptus, Openstack, Opennebula etij.

c. **Hybrid Cloud**. Cloud hybrid është një kombim i dy ose më shumë infrastrukturave Cloud si p.sh. Public, Privat ose Community të cilat janë të dallueshme nga njëra-tjera por lidhen bashkë nga një teknologji e caktuar që i lejon të dhënave dhe aplikacioneve të jenë të lëvizshëm [21]. Nëpërmjet Hybrid Could, shkëmbimi i burimeve bëhet më i lehtë dhe i mundshëm sipas nëvojave. Një mjedis hibrid është zgjedhja më e mirë kur kompania dëshiron të përdorë një aplikacion SaaS por ka shqetësime për sigurinë. Ofruesi i Shërbimit mund të krijojë një Private Cloud për një kompani brënda firewall-it të tyre. Kjo krijon mundesinë e ofrimit të një rrjeti privat virtual (Virtual Private Netëork - VPN) i cili rrit sigurinë dhe ofron akses të kontrolluar.

d. **Cloud community**. Në këtë rast, infrastruktura Cloud ndahet nga disa organizata dhe mbështet një komunitet specifik që ndan shqetësime të njëjta (p.sh. misioni, kërkesat e sigurisë, politikat dhe konsideratat e pajtueshmërisë)[21]. Mund të menaxhohet nga organizatat ose një palë e tretë. Kjo tip infrastrukturë ngrihet mbi baze bashkëpunimi organizatash te ndryshme për vendosjen e aplikimeve të njëjta.

2.4 Teknologjitë mundësuese të Cloud Computing

Ka dy teknologji që bëjnë të mundur ndarjen e burimeve dhe në këtë rast shërbejnë për të krijuar Cloud Computing.

Balancimi i ngarkesës (Load Balancing- LB)

Me balancim të ngarkesës është çelësi i suksesit për arkitekturat e Cloud. Arësyet e përdorimit të balancimit të ngarkesës mund të jetë një mbingarkesë në memorie, procesori CPU, rrjeti ose vonesa në ngarkesë e shaktuar nga një mbingarkesë kërkesash për shërbim. Sipas Mac Vittie, balancimi i ngarkesës ka të bëjë me aftësinë e shpërndarjes së procesit të punës në mënyrë të barabartë midis 2 ose më shumë kompjuterëve që ofrojnë një shërbim të caktuar. Në mënyrë që burimet të përdoren në mënyrë efikase dhe si rrjedhojë të rritet performanca dhe disponueshmëria për të përballuar ngarkesën [22]. LoadBalancer-at janë makina virtuale të specializuara që kryejnë vetëm funksionin e balancimit të ngarkesës drejt serverëve që janë të konfiguruar për të ofruar shërbim nëpërmjet këtij shpërndarësi ngarkese.

Një balancues ngarkese është i aftë të përballojë sasi të ndryshme kapacitetesh pune duke përshtatur vendimet e shpërndarjes sipas momenteve kur bëhet një kërkesë. Kjo është një zgjidhje e balancimit të ngarkesës që përdoret shpesh në shërbimet e internetit, ku ideja e balancimit të ngarkesës drejtohet nga një aplikacion [22] . Pra me pak fjalë qëllimi i balancimit të ngarkesës është: Përmirësimin e performancës, Ruajtjen e Stabilitetit të Sistemit, Ndërtimi një i sistemi tolerant ndaj gabimeve dhe ku ndryshimet në të ardhmen realizohen pa ndryshime të mëdha infrastrukturore e llogjike[23].

Ka dy lloj algoritmesh për Load Balancer:

- **Algoritem Statik**: Në ketë lloj algoritmi ngarkesa ndahet në mënyrë të barabartë midis serverëve. Ky algoritëm ka domosdoshmëri për të njohur burimet e sistemit që vendimet e zhvendosjes së ngarkesës të mos varen nga gjendja e sistemit.
- **Agoritem dinamik**: Në këtë lloj algoritmi serveri me ngarkesë më të vogël në të gjithë rrjetin ose sistemin kërkohet ose preferohet si balancues ngarkese.

Duke qenë pjesë tepër e rëndësishme e Cloud Computing si shërbim i domosodoshëm dhe që ofron facilitete që përdoren në masë, nga ofruesit e Cloud i është vënë një theks tepër i madh monitorimit në detaj të këtyre burimeve.

2.5 Virtualizimi

Virtualizimi është një teknologji tjetër e përdorur që në vitet 1960 kur kishte probleme me formën tradicionale e qendrave të dhënave dhe kërkohej një zgjidhje më dinamike [28]. Sic u theksua edhe më parë në këtë studim, virtualizimi është një metodë që përdoret gjerësisht në sistemet Cloud për të ndarë burimet e ndryshme si për shembull memorien, severin apo rrjetin, duke përmiresuar kështu përdorimin e tyre, gjithashtu duke ulur kostot në infrastrukturën IT dhe duke rritur abstraksionin.

Karakteristikat e Virtualizimit: Sipas Gerald J. Popek dhe Robert P. Goldberg [24], një makinë virtualizuese duhet të ketë këto tre lloj karakteristikash:

Ekuivalenca ï çdo program që është ekzekutuar ne një makinë virtuale (VM) duhet të shfaqë sjellje të njëjtë me atë që programi do të jepte;

Efikasiteti - ekzekutimi realizohet në mënyrë të automatizuar pa ndërhyrjen e operatorit;

Kontrolli i burimeve - asnjë program nuk mund të ekzekutohet në burime të cilat nuk i janë caktuar makinës virutale në mënyrë të qartë dhe specifike.

Gjithashtu, në teknologjinë e virtualizimit ka një pjesë softuare që lejon ndarjen e serverit fizik me instanca të shumëfishta të makinave virtuale dhe që njihet si Hypervisor. Hypervisor është përgjegjës për supervizimin dhe ambientin që makinat komunikojnë, ndajnë burime dhe realokohen sipas kërkesave. Kjo lidhje midis pjesës virtuale dhe fizike është shumë e rëndësishme në mjedisin Cloud sepse krijon dinamizëm. [29]

Kemi disa lloje të virtualizimit në Sistemet Cloud, ndër to mund të përmendim Virtualizimin e Memories, aplikacionit, rrjetit, dhe serverit. Të gjitha këto janë bërë kryesisht për të thjeshtuar punën e përdoruesve, duke u bërë të mundur të punojnë pa qenë të kufizuar ne kohe apo distance. Gjithashtu, këto sisteme mundesojnë përdoruesve që të ndajnë të njëjtat burime në të njëjtën kohë pa interferuar me njëri-tjetrin, duke rritur kështu nivelin e performancës dhe abstraksionit.

- Në virtualizimin e memories kemi një strukturë kryesore që quhet ñShadow Page Tableò. Kjo faqe memorieje ndodhet në sistemin e shfrytëzimit dhe ruan një tabelë faqesh që përkthehen nga faqe virtuale në faqe fizike. Në sistemet virtuale të

shfrytëzimit, faqet fizike shikohen nga makina virtuale dhe konsiderohen gjithashtu si faqe virtuale.

- Virtualizimi i Serverit është një nga llojet më fitimprurëse të virtualizimit duke marrë parasysh rolin që luan në fushën e Teknologjisë se informacionit. Ky lloj virtualizimi konsiston në krijimin e serverave virtual në një server fizik me ndihmën e një softueri të aftë për tu virtualizuar [25].
- Virtualizim i plotë, konsiston ne izolimin e serverave virtual nga njëri tjetri që punojnë në te njëjtën makinë fizike, në menyrë që makinat virtuale mos të jenë në dijeni të veprimeve të njëra tjetrës. Kjo gjë bëhet nga Hypervisor, i cili bën ndarjen e punës dhe ua dëgron serverave virtualë në mënyrë që mos të jenë në dijeni të njeri-tjetrit.
- Virtualizim paravirtual, ndryshe nga virtualizimi i plotë lejon që makinat virtuale të jenë në dijeni të punës së njëra-tjetrës, dhe si rrjedhojë Hypervisori nuk ka shumë punë.
- Virtualizim në nivel OS, në këtë nivel nuk është i nevojshëm hypervisori pasi këtë rol e kryen vete sistemi i shfrytëzimit. Sfida që ka në këtë mes është pavarësia e makinave virtuale me njëra tjetrën, duke qenë se sistemi i shfrytezimit është i njëjti për të gjitha makinat.

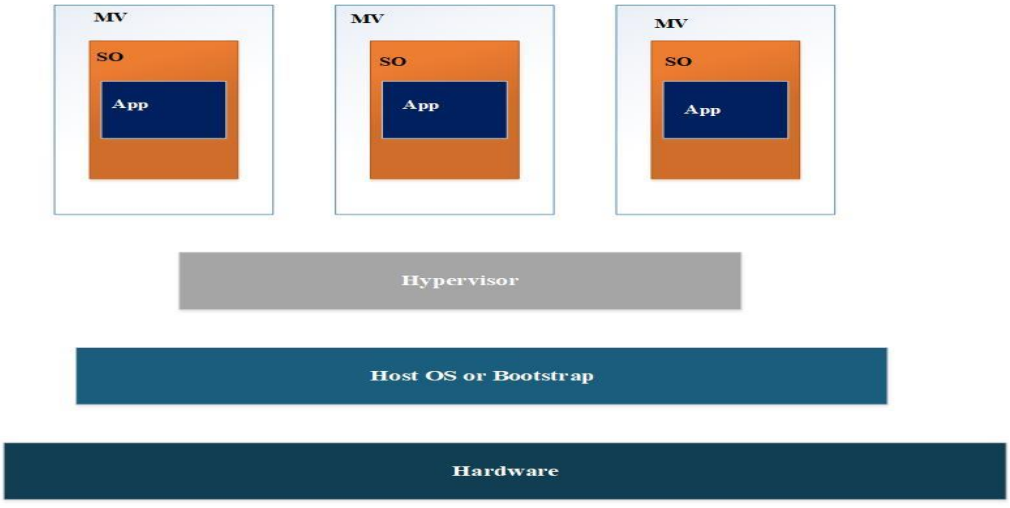


Figura 2 2: Virtualizimi në Cloud

Kur flasim për vitualizimin, duhet të kemi parasysh disa karakteristika të rëndësishme që janë: ndarja (partitioning), izolimi (isolation) dhe kapsulimi (encapsulation). Ndarja i referohet përdorimit të burimeve të një sistemi fizik të vetëm nga shumë aplikacione apo sisteme shfrytëzimi në të njëjtën kohë. Karakteristika e dytë i referohet izolimit të makinave virtuale nga sistemi fizik apo makinat e tjera. Në këtë menyrë do të kuptojmë që të dhënat nuk ndahen ndermjet makinave, gjithashtu nëse një nga makinat virtuale pëson defekt, mos të ndikojë tek makinat e tjera.

Kapsulimi është karakteristika që e shfaq një makinë virtuale si një entitet të vetëm, dhe mund të identifikohet lehtesisht në bazë të shërbimit që ofron. Ai mbron aplikacionet duke bërë të mundur që këta të fundit mos të interferojnë me njëri-tjetrin.

Virtualizimi ka për qëllim permbushjen e disa objektivave, të cilat janë:

- Të jetë kompatibël, pra, të mbështesë sistemet e shfrytëzimit të pamodifikuara si sistemet e vjetra.
- Performanca, të ketë performance të mirë duke përdorur teknika që mos të ngarkojnë me óverheadò.
- Me pak kompleks, të jetë relativisht i thjeshtë, mos të ketë shume kod, si rrjedhojë të ketë më pak dobësi.

Një nga sistemet më të suksesshme që disiplina e shkencave kompjuterike njeh, është Java Virtual Machine [26]. Ky shembull virtualizimi mundëson ekzekutime të gjuhës Java pa marrë parasysh arkitekturën (ISA) e përdorur nga një set harduare i caktuar. Kjo ndodh për shkak se kodi i Java ekzekutohet në një ambjent virtual.

2.6 Avantazhet e Virtualizimit

Virtualizimi ka disa avantazhe që do ti përmendim shkurtimisht [30]:

- Virtualizimi lejon ndarjen e sistemit operativ nga harduare. Falë virtualizimit tashmë nuk do të jetë e nevojshme që pjesa softuare të komunikojë direkt me pjesën

harduare, por me një harduere të virtualizuar. Kjo metodë gjithashtu shfaq një pamje të unifikuar të harduare, sido që të jetë harduare, virtualizimi lejon një ndërfaqe harduerë unike, dhe kjo funksionon për harduerë të ndryshëm. Qëllimi është që makina të ndryshme fizike me varietete pajishjesh Input/Output të jenë njësoj në dukje.

- Virtualizimi lehtëson migrimin live të makinave virtuale (live migration of VM), me anë të së cilit mund të zhvendosim lehtësisht një makinë virtuale nga një server fizik në një tjetër live. Me fjalë të tjera, nuk është e nevojshme ta ndalojmë dhe startojmë sërish makinën virtuale, dhe si rrjedhojë kemi rritje të performancës, menaxhon lehtësisht difektet pa nderprerë shërbimin dhe është më e lehtë për tu mirëmbajtur.
- Kostoja e menaxhimit dhe energjisë është një tjetër e mirë që ofron virtualizimi nëpermjet konsolidimit, pra nuk mban shumë servera në punë, secili ka makinat e veta virtuale, pra, i centralizon këto makina virtuale në pak servera fizike kështu që ka një ulje të shpenzimeve të energjisë por gjithashtu ulje të nivelit të kompleksitetit të kabllimit në dhomat e serverëve.
- Avantazhi tjetër është në ndalimin (suspend) dhe vazhdimin (resume) e makinave virtuale, është shumë fleksibël në menaxhimin e tyre . Gjithashtu ofron ruajtje të gjendjes së makinës virtuale (checkpoint/snapshot) dhe kthim pas në të, në këtë mënyre ruan gjendjet në një pikë të caktuar në menyrë që të kthehesh lehtesisht për të gjetur dhe riparuar gabimet.
- Kjo metodë ofron besueshmëri të lartë duke ofruar izolim midis aplikacioneve. Pra, aplikacionet në kete rast mund të ndodhen në një server fizik, por në makina virtuale të ndryshme. Me këtë rast, është më se e sigurtë që nëse një virus infekton një makine virtuale, e ka të pamundur të infektojë makinat e tjera.
- Mekanizmat e sigurisë: në sisteme shfrytezimi pa virtualizim, ekzekutimi bëhet brenda sistemit, ndërsa virtualizimi ofron ekzekutim jashtë makinës virtuale ose në një makinë tjetër virtuale (ose nivel hypervisor) për të rritur nivelin e sigurisë.

Virtualizimi në Platformën Cloud

Në platformat Cloud, virtualizimi i plotë është teknika më e përdorur ndërsa virtualizimi i pjësëshëm përdoret për të përmirësuar performancën e virtualizimit. Për shëmbull, Amazon Web Services (AWS) më parë ka përdorur Xen [95], një hypervisor si një makine virtuale monitoruese (Virtual Machine Monitor -VMM) e cila e ndihmon në krijimin dhe ekzekutimin e disa makinave virtuale me një kosto të vogël. Përdorimi i një virtualizimi të plotë, Xen krijon mundësinë e një izolimi të plotë midis disa makinave virtuale (VMs) dhe i lejon këtyre makinave të ndajnë burimet virtuale hardëare në mënyrë të sigurtë. Tashmë ata përdorin një ñhypervisorò që mbështetet në KVM [96] në instancat e reja të tipit C5 [97]

Me teknologjinë e virtualizimit, grupimi i makinave fizike (PM) brenda një qendre të dhënash (Datacenter) Cloud mund të mbështesë një numër të madh makinash virtuale (VM) dhe krijon një administrim të centralizuar të të gjithë makinave virtuale. Në qendrën e të dhënave Cloud, një makine virtuale (VM) mund të kontollohet në mënyrë më fleksibël se sa një makinë fizike sepse konfigurimi i makinës virtuale duke përfshirë edhe numrin CPU, përmasat e memories, kapaciteti I/O mund të ndryshohen dhe rregullohen lehtësisht, makina virtuale mund te shtohet ose hiqet sipas nevojave, makina virtale mund të lëvizet nga një makinë fizike tëk një tjetër më një kosto të ulët dhe të gjithë makinat virtuale mund të ndajnë kapacitet e paisjeve në dhomën e serverëve ku është implememtuar Cloud. Pra si rezultat, makina virtuale në mjedisin Cloud mund të arrijë një shfrytëzim të lartë të burimeve, një disponueshmëri të lartë dhe kosto të ulët të burimeve.

KAPITULLI III

Elasiciteti, Konceptet dhe Arkitekturat

3.1. Hyrje

Krijimi i pajisjeve si telefona celularë, tablet, PC, laptop, pajisje të specializuara si internet radio, pajisje meteorologjike, GPS, ñtë afta për të lundruar në internetò është shoqëruar dhe me disa vështirësi duke filluar që nga rritja e kompleksitetit në programim, përcaktimin e infrastrukturave mbështetëse, implementimin e kontrolleve të sigurisë, rritje e prezencës në internet për përdoruesit ku shtimi i shërbimeve dhe pasurimi i tyre janë si një nga qëllimet kryesore në vetvete të këtyre sistemeve (p.sh e-baking-Banka online 24 ore). Shtimi i shërbimeve ñonlineò kërkon që dhe këto sisteme të jenë të aksesueshme 24 orë në ditë (sidomos sistemet globale, Facebook, Twitter etj). Rritja e kohës së të qenit ñonlineò është shoqëruar me një rritje te padurimit të përdoruesve, kundrejt shpejtësisë së përgjigjes së aplikimeve si dhe disponueshmerisë së tyre. Përdoruesit thjesht duan të marrin shërbimin e kërkuar pa vlerësuar sistemet, numrin e aplikimeve, serverat, lidhjet e internetit etj.

Është një detyrë e ofrueseve të shërbimeve Cloud për të zgjidhur këto problematika si dhe për tõu ofruar klientëve te tyre mundësi fleksibel për rritjen apo zvogëlimin e burimeve qe ata do te marrin në përdorim klientët dhe një faturim sa me fleksibel për përdorimin e burimeve të Cloud. Burimet në Cloud që do të duhen tõu ofrohen përdoruesëve do të duhet të jenë të shkallëzueshme dhe elastike njëkohësit, çka do të thotë që përballimi i ngarkesave te mëdha si dhe realizimi/plotësimi i kënaqshëm i kërkesave të paraqitura në aplikime. Pikërisht më poshtë do të shpjegojmë më në detaj elasticitetin.

3.2 Cfarë është Elasticiteti

Në literaturë ka pasur shumë përkufizime për elasticitetin [32],[33],[34],[35]. Sipas [36], ñ Elasticiteti është shkalla në të cilën një sistem është në gjendje të përshtatet me ndryshimet e ngarkesës së punës duke siguruar zvogëlimin ose zmadhimin e burimeve në një mënyrë autonome, në mënyrë të tillë që çdo kohë burimet e disponueshme përputhen me kërkesën aktuale dhe kjo përputhje është sa më përafërtò.

Sipas këndvështrimit të studimeve të lartpërmendura dhe në mënyrë më praktike, elasticiteti përbën aftësinë e një sistemi për të shtuar dhe hequr burimet (të tilla si bërthamat e CPU-së, memoria, makinat virtuale (VM) dhe kontejnerët e instancës) për t'iu përshtatur ndryshimit të ngarkesës në kohë reale. Elasticiteti është një karakteristikë thelbësore për Cloud Computing. Njihen dy lloje tipe elasticiteti: vertikal dhe horizontal. Me elasticitet vertikal kuptojmë rritjen ose zvogëlimin e karakteristikave të burimeve si p.sh shepjtësia e CPU, numri i CPU, memoria RAM ose haësirë në disk, gjërësia e brezit, pra elasticiteti vertikal ka të bëjë me ndryshimet e parametrave të makinave virtuale ekzistuese. Me elasticitet horizontal kuptojmë shtimin ose heqjen e instancave të burimeve që lidhen me aplikacionet, pra që nënkupton me krijimin e makinave të reja virtuale me të njëjtët ose me oarametra më të mëdhenj se makina virtuale fillestare[38]. Pikërisht rikonfigurimi i makinave virtuale në bazë të elasticitetit vertikal ose horizontal paraqiten në figurën e mëposhtme. Duhet të përmendim që për trajtuar këtë cështje ofruesit e Cloud përdorin Load Balancer.

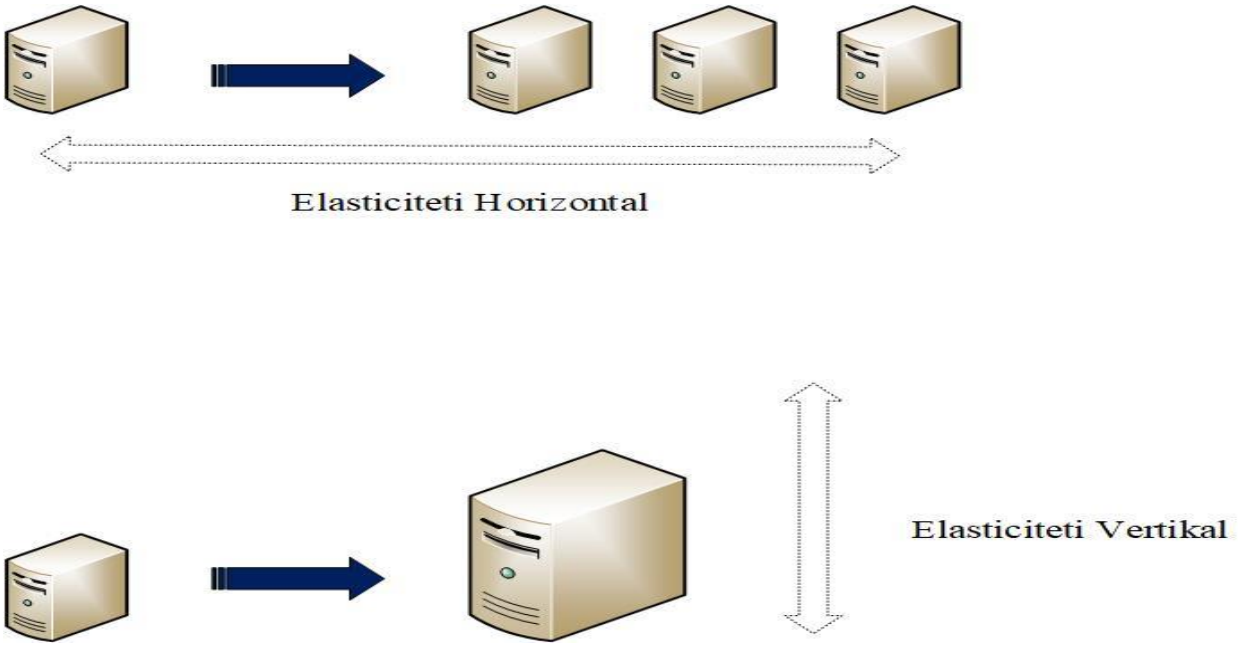


Figura 3 1 Elasticiteti Horizontal dhe Vertikal

Gjithashtu ka edhe terma të tjerë të cilat lidhen me elasticitetin siç janë shkallëzueshmëria dhe eficienca por kuptimi i tyre është i ndryshëm pavarësisht se ato përdoren në mënyrë të ndërsjelltë në disa raste. Ka dy qasje në implementimin e politikës së elasticitetit. Qasja e parë ka të bëjë me qasjen e kontrollit teorik (control theoretical approach) ku vendimi për alokimin e burimeve bazohet tek performanca e aplikimit dhe projeksioni i tij bazohet tek Cilësia e Shërbimit (QoS). Qasja e dytë quhet Heuristic Rule-based Approach ku vendimi për shkallezueshmërine bazohet mbi specifikimet tek një grup rregullash ñheuristicò me bazë parashikime ose shfrytëzimi i teorive probabilitare ose interligjencës artificiale.

Shkallëzueshmëria është aftësia e sistemit të mbështesë rritjen e ngarkesës së punës duke përdorur burime shtesë[37]. Pra shkallëzueshmëria është e pavarur nga koha dhe koha nuk ka efekt ne system. Për të kuptuar më mirë, më poshtë është dhënë një ekuacion i cili përmbledh të dhënat e elasticitet në Cloud Computing.

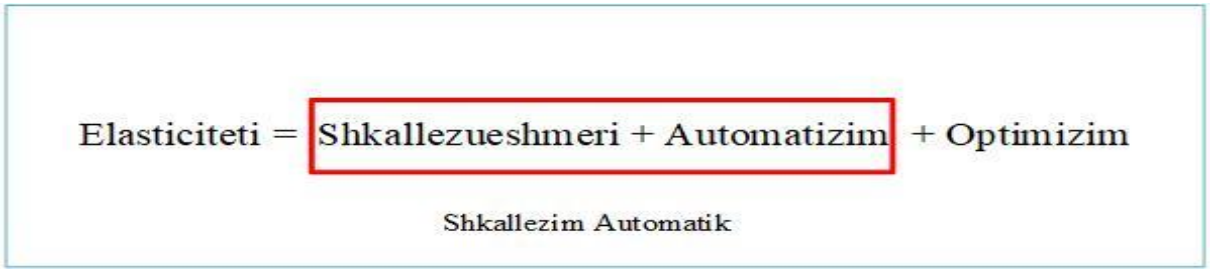


Figura 3 2 Ekuacioni i Elasticitetit në Cloud (referuar studimit [39])

Pra, elasticiteti është një cilësi dinamike sepse i lejon sistemit të rritet ose zvogëlohet sipas sistemit operativ, ndërsa shkallëzueshmëria është statike.

Termi tjetër që lidhet elasticiteti, është me efikasitetin, i cili ka të bëjë me mënyrën se si burimet e Cloud shfrytëzohen në mënyrë efikase gjatë rritjes ose zvogëlimit të burimeve. Pra ky koncept tregon sasinë e burimeve të konsumuara për një sasi të caktuar të pune dmth: sa me e ulët është sasia aq më e lartë është efikasiteti i sistemit [36]. Në ndryshim nga elasticiteti, efikasiteti nuk është i kufizuar në llojet e burimeve që janë shkallëzuar si pjesë e mekanizmave të adaptimit të sistemit. Efikasiteti mund të ndikohet nga faktorë të tjerë të pavarur nga mekanizmat e elastik të sistemit.

Është e rëndësishme të theksohet që shkallëzimi lart ose poshtë i burimeve çon në ndryshime të burimeve të alokuara nga burimet e kërkuara. Pikërisht, dy faktorë të rëndësishëm në lidhje me saktësinë e sistemit elastik: mbi-përdorimi ka të bëjë kur burimet e ofruara (O) janë më të mëdha se sa ato të kërkuara (K) dhe nënpërdorimi përbën të kundërtën. Pra, në rastin e parë kur O>K, kemi kosto ekstra për shkak të burimeve të pa nevojshme. Kur K>O, kjo çon në keqësim të performancës dhe prishe të Marrwveshjes pwr Nivelin e Shwrbimit (SLA).

3.3 Klasifikimet

Klasifikimin e zgjidhjeve të elasticitet do ta bazojmë në studimet e ndryshme që janë analizuar [39], [40], [41] karakteristikave kryesore. Gjithsesi, sic është propozuar nga Naskos, Gounaris dhe Sioutas, ku studim do të konsiderojë 6 karakteristika për zgjedhjet e

elasticitetit (1)Fushëveprimi (scope); (2) Qëllimi (3) Vendim-marrja (4) Veprimet e Elasticitetit (5) Ofruesi dhe (6) Vlerësimi [49].

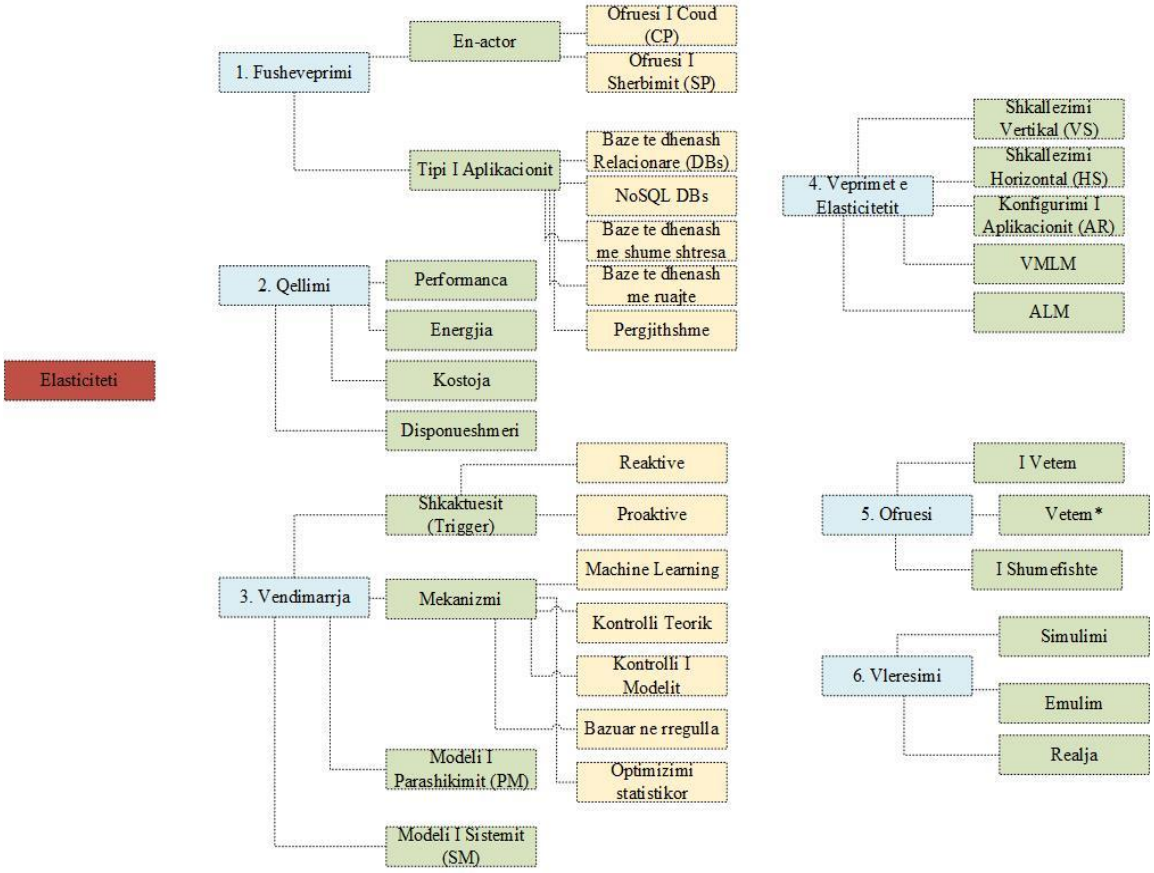


Figura 3 3 Karakteristikat e Elasticitetit (referuar studimit [49])

Fushëveprimi (scope) që është karakteristika e parë sipas studimeve të lartëpërmëndura, përcakton se ku kontrollohen veprimet e elasticitetit. Fushëveprimi është ndarë në kategori klasifikuese: 1. ñEnactorò që tregon nëse teknika e elasticitetit aplikohet nga ofruesi ose përdoruesi dhe 2. Tipi i aplikacionit i cili i referohet mënyrë së si elasticiteti është përdorur në disa lloj aplikacionesh në Cloud: Bazë të dhënash relacionale, Bazë të dhënash NoSQL, aplikacione më shumë shtesa dhe të përgjithshme. Përgjithësisht, IaaS Cloud ka një kontrollues elasticiteti i cili është përgjegjës për konvertimin e kërkesave të përdoruesve në veprime të ofruesve. Kontrolluesi përdor monitorimin e të dhënave nga aplikacionet dhe merr vendimeve nëse burimet duhet të shkallëzohen apo jo [42] .

Qëllimi (Purpose): Nëpërmjet kësaj karakteristike klasifikohen teknikat në varësi të qëllimit të veprimit të elasticitetit. Qëllimi mund të jetë një nga këto: Performanca, Disponueshmëria, Kostoja, dhe Energjia.

- Performaca, me qëllim përmirësimin e saj
- Disponueshmëria me qëllim rritjen e disponueshmëria
- Kostoja, me qëllim një menaxhim sa më efektiv i kostove
- Energjia, me qëllim kursimin e energjisë

Vendimmarrja : Në lidhje me vendimmarrjes, autorët [39] kanë sugjeruar 4 kritere të ndryshme që mund karakterizojnë vendimmarrjen

1. Shkaktuesi (Trigger) që tregon nëse mekanizmi i elasticitetit është shkaktuar në mënyrë reaktive ose proactive

2. Mekanizmi i cili i referohet metodologjisë së marrjes së vendimeve

3. Modeli i parashikimit (Prediction Model ï PM) që tregon përdorimin e modelit për të parashikuar ngarkesa në hyrje variable ose vlera specifike të matshme në të ardhmen

4. Modelimi i Sistemit (SM) i cili i referohet përdorimit të modelit që prezanton sjelljen elastike të sistemit mbi të cilën ndërtohet e gjithe politika e elasticitetit.

Veprimet e elasticitetit: Burimet e Elastike të Cloud mund të aplikohen në forma të ndryshme dhe mund ti referohen modifikimit (i) përmasave/ Shkallëzim Vertikal (VS) I jep më fleksibilitet sistemit cloud që të operojë me ngarkesa të ndryshme. (ii) Vendodhjes (VM Migrantin (VMLM) - ose (iii) numri makinave virtuale (VM) të përdorura (Shkallëzim Horizontal, HS).

Gjithasht Veprimet e Elasticitetit përfshijnë edhe dy tipesh (IV) rikonfigurimi i aplikacionit ku mjeti elastik është i aftë të trajtojë aspekte të ndryshme të aplikacionit (V) Migrimi i Drejtëpërdrejtë i Aplikacionit (ALM) ku komponentët specific të aplikacionit migrojnë në vend të të gjithë makinës virtuale

Ofruesi: Kjo kategori klasifikuese i referohet numrit të ofuresve të infrastrukturës Cloud që mjeti elastik mbështet njëkohësisht. Pra, zgjidhja elastike mund te aplikohet ne një ofruesi Cloud ose në disa prej tyre. Kjo do të thotë që vlerat e mundshme janë (i) Vetëm, që tregon se vetëm një ofrues cloud mbështetet (ii) Vetëm*, që tregon se

mbështeten më shumë se një ofrues por jo njëkohësisht dhe (iii) të shumëfishtë, ku është kontrolli i elasticitetit është i përhapur midis gjithë ofruesve të shumëfishtë njëkohësisht [43]. Pjesa më e madhe propozimeve dhe zgjidhjeve për elasticitetin fokusohen tek ofruesi i vetem i Clould.

Vlerësimi: Ky aspekt i referohet vlerësimit të çdo lloj punë. Vlerat e mundshme janë: (i) Simulimi, ku rezultatet merren nga përllogaritjet në një mjedis artificial të simuluar (p.sh. (OMNeT ++), (ii) Emulim ku rezultatet e vlerësimit merren nga një mjedis artificial që sillet sipas botës reale, dhe (iii) Realja, ku mjeti elastik zbatohet në një infrastrukturë reale Cloud.

3.4 Elasticiteti i IaaS

Elasticiteti aplikohet tek të gjithë modelet e shërbimit të Cloud Computing. Tek modeli IaaS fokusi i shkallezueshmërisë është tek niveli i aplikacionit, pra numri i përdoruesve të cilët përdorin shërbimet dhe softuare-et që ekzistojnë. Tek modeli IaaS, elasticiteti arrihet tek niveli i harduare fizik dhe virtual. Tek ky model, burimet harduerë janë ofruar si shërbime për përdoruesit që të drejtojnë dhe menaxhojnë aplikacionet e tyre. Gjithashtu, IaaS Cloud nuk kërkon ndryshime të mëdha në arkitekturë ose programim dhe kjo i lejon organizatat të fokusohen tek avantazhet e tyre kryesore në garën e biznesit. Ky lloj aplikacioni sjell benefite të mëdha sepse organizatat mund të aksesojnë burime masive duke paguar kostën e orës së përdorimit pa parapaguar ose pa angazhime afatgjata, dhe së dyti organizatat marrin burime elastike në baze të kërkesës. Ky studim do të fokusohet tek modeli IaaS Cloud duke qënë se përmban conceptin e Elasticitetit.

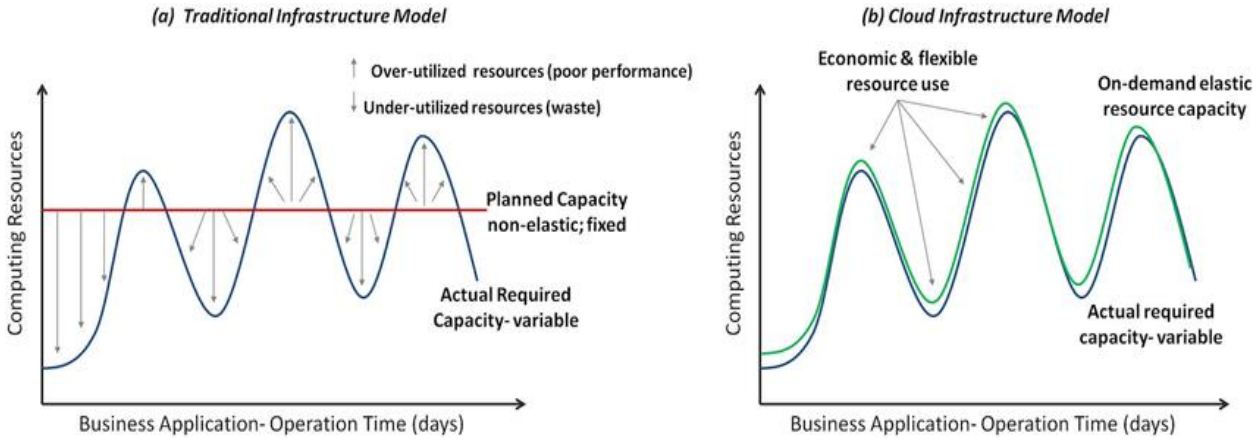


Figura 3 4 Elasticiteti dhe Efektivitetin e Kostos në IaaS Cloud (përshtatur nga [44])

Tradicionalisht organizatat kanë planifikuar infrastrukturën informatike bazuar në kapacitetin e tyre maksimal të pritshëm për kapacitet e burimeve siç paraqitet tek figura e mësipërme (3.4). Duke patur parasysh dinamikën dhe shpejtësinë e ndryshimeve, oraganizatat kanë nevojë për rritje dhe planifikimi i kapaciteteve duhet të jetë më fleksibël dhe më ekonomik për dy arsye: Së pari, planifikime të tilla të kapaciteteve përfshin investime shumë të mëdha kapitali dhe ka një periudhë shumë të ngadaltë në kthimin e tyre. Se dyti, një kapacitet kaq i madh informatik nuk mund të përdoret në mënyrë efikase. Siç tregohet në figurë mund të ketë periudha kohore kur burimet janë të pa shfrytëzuara. Gjithashtu, siç e dimë, ka kosto mirëmbajtja e infrastrukturës dhe ka periudha kur aplikacionet e tejkalojnë kapacitetet e planifikuara. Modeli IaaS i krijon mundësinë përdoruesve të arrijnë eficiencë dhe fleksibilitet në përdorimin e burimeve infrastrukturore siç mund të shihet në pjesën (b) të figurës. Në këtë rast, skenaret e nën-shfrytëzimit dhe mbi-shfrytëzimit mund të trajtohen nga organizatat në mënyrë eficientë nëpërmjet ofrimit elastik të burimeve në modelin IaaS. Pra në këtë model planifikimi tradicional nuk është i nevojshëm.

3.4.1 Arkitektura dhe Mekanizmi i Elasticitetit IaaS

Ofruesit e IaaS bashkojnë burimet si p.sh. njësia procesuese, hapësira e diskut, gjerësia e rrjetit në paketën e shërbimit. Këto paketa ofrohen si instanca shërbimi duke ndryshuar

një ose disa burime ose kapacitete. Ofruesit e IaaS ofrojnë nivele të ndryshme detajimi që i lejon përdoruesëve të përshtasin burimet e Cloud. Më poshtë po japim disa pika të përbashkëta për shkallëzimin e burimeve.

- Vendosja/ heqja e serverave fizik ose virtual me kapacitete të njëjta ose më të mëdha
- Rritja ose zvogëlimi i burimeve si p.sh. CPU, memoria, hapësira e diskut duke shtuar ose hequr/shtuar componente hardëare në makina ekzistuese
- Rritje/zvogëlimi i shpejtësisë së rrjetit dhe sasitë e adresave IP
- Rritja/zvogëlimi i sasisë së transferimit të të dhënave dhe numri i operacioneve/kërkesave të të dhënave të burive Cloud

Në varësi të paketës, ofruesit e IaaS ndryshojnë në varësi të mënyrës se si e mundësojnë elasticitetin dhe të shkallëzueshmërisë së burimeve. Ka dy tipe arkitekturash shkallëzueshmërie që përdoren dhe që u përmenden edhe me lartë.

3.5 Aplikime multishtresorë në Cloud

Një aplikacion Cloud mund të jetë një aplikacion infrastrukture ose aplikacion që aksesohet nga përdoruesë përfundimtarë. Aplikacionet e infrastrukturës kanë një strukturë të thjeshtë me një ose dy shtresa. Por nga ana tjetër, strukturat me përdorues përfundimtarë (end-user) janë më komplekse. Këto aplikime mund të shtrihen në shumë shtresa. Pra, më poshtë po paraqesim një shembull se si shtresat bashkëpunojnë me njëra tjetrën për të trajtuar kërkesat nga përdoruesit përfundimtarë. Pikërisht në këtë shembull kemi një Load Balancer që siç e kemi përmendur ndihmon në ndarjen e ngarkesës së punës midis nyjeve të sistemit në mënyrë që të përmirësohet shfrytëzimi i burimeve dhe për të arritur performancën më të mirë ku të gjithë kërkesat e përdorueseve të plotësohen. Në këtë shembull siç shihet në figurën 3.5, Load Balancer merr kërkesa dhe i transmeton ato tek Apache HTTP server për të trajtuar kërkesat HTTP. Apache përdoret gjerësisht në serverat web në të gjithë botën. Kërkesat e Apache mund të mbështeten tek kërkimi i bazës së të dhënave, modifikimi i të dhënave ose futje e të dhënave të reja. Amoeba LB [76]

server shpërndan kërkesat e bazës së të dhënave tek serverat e bazave të dhënave MySQL [98] duke bërë të mundur kërkimin, futjen dhe azhornimin e tabelave të bazës së të dhënave. MySQL është një bazë e dhënash popullore që përdoret për të ruajtur të dhënat dhe informacionin e konfigurimet të aplikacioneve në web. Pikërisht, MySQL si një pjesë e sistemeve të bazave të dhënave që egzistojnë në Internet ka bërë të mundur zhvillimin e mëtejshëm të faqeve të internetit dinamike [45] . Baza e të dhënave MySQL ruan të gjithë të dhënat që përfaqësojnë shërbimet e aplikacionit tonë.

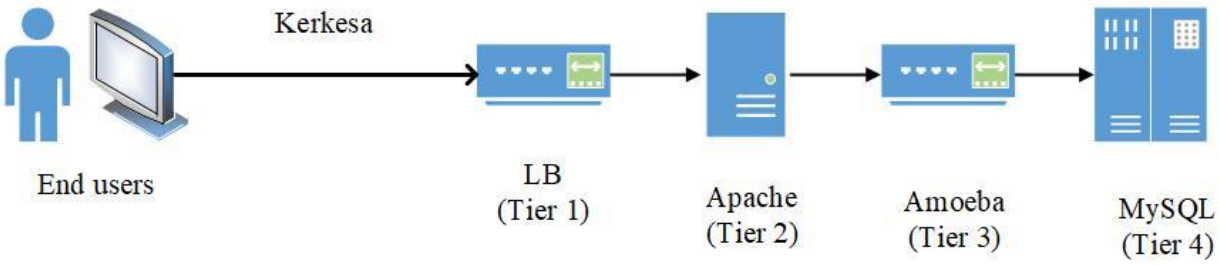


Figura 3 5 Arkitektura Multi Shtresore

Në aplikacionet më shumë shtresa, serverat kategorizohen me shtresa të ndryshme duke u bazuar tek funksionaliteti. Pra sic duket edhe nga figura, Serverat në dy shtresa LoadBalancer si LB(në vetvete) dhe Ameoba shpërndajnë kërkesat tek shtresa e shërbimeve ose ruajtes. Në këtë shembull, kërkesat dërgohen tek serveri i shërbimit Apache i cili është përgjegjës për trajtimin e kërkesave HTTP dhe implementimin e logjikës së biznesit. Serverat tek niveli i ruajtes si në rastin konkret baza e të dhënave MySQL përdoret për të menaxhuar të dhënat e aplikacionit.

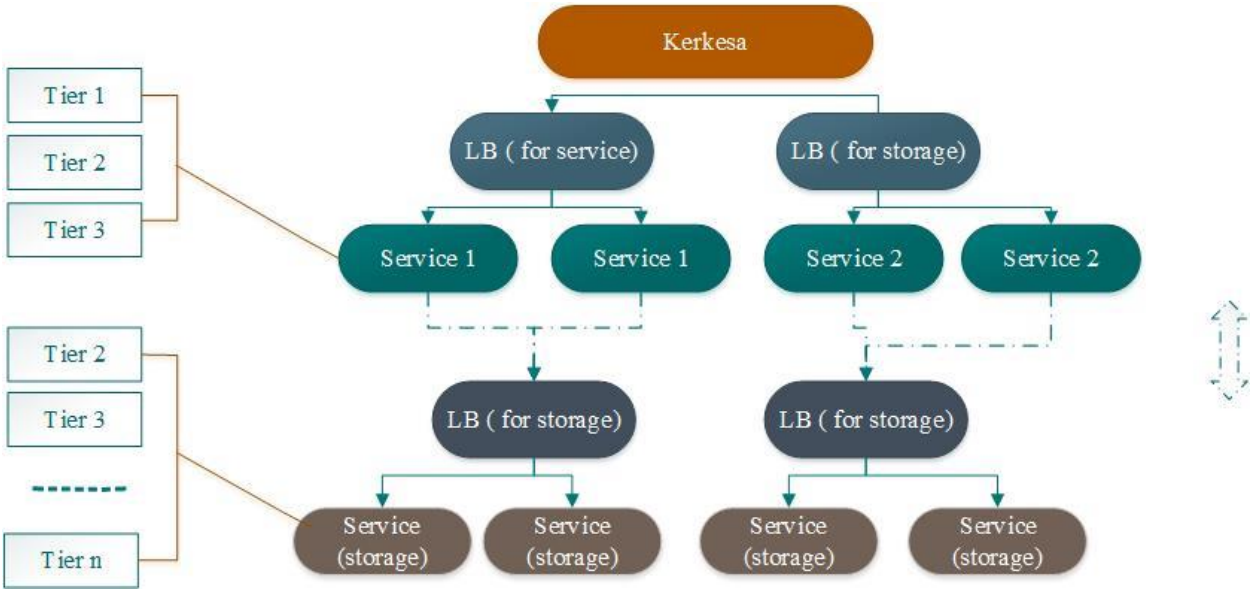


Figura 3 6 Renditja Vertikale e Shtresave në Cloud

Çdo aplikacion ka të specifikuar një sërë kërkesash dhe kufizimesh që mund të jenë të specifikuara nga ofruesi i shërbimit tek Marrëveshja e Shërbimit (SLA). Një kërkesë për Cilësinë e Shërbimit (QoS) përcaktohet nga koha e kërkuar për ti kthyer përgjigje një kërkese. Kjo kohë e kërkuar për përgjigjen tregon vonesën maksimale të pranueshme për përgjigjen kur një kërkesë përshkon aplikacionin; që është, intervali kohor midis momenteve të mbërritjes dhe daljes së kërkesës. Kjo kohë e kërkuar mund të jetë një mesatare e kohës së përgjigjes për të gjitha kërkesat ose përqindja më e lartë e shpërndarjes së kohës së përgjigjes. Kufizimi i kostos përbën totalin e buxhetit për vendosjen dhe përdoriminin e aplikacionit. Përveç kësaj, secila shtresë ka një kufizim të burimeve që kufizon numrin maksimal të serverave në këtë shtresë.

Një aplikacion me shumë shtresa përbëhet nga dy pjesë: (1) grupi i serverit S përfshirë të gjithë serverat e aplikacionit dhe (2) grupi i kërkesave D që prek të gjithë specifikimet e përcaktuara në SLA.

Arkitektura me shumë shtresa garanton modularitetin e aplikacioneve Cloud dhe lehtëson kontrollin e shtresave.. Secili server i përcaktohet një shtresë ID unike për të qenë lehtësisht i dallueshëm dhe integrueshëm në cdo sistem. Për shembull, në figurën e ilustruar

më sipër , ID-të ë shkallëve nga LoadBalancer deri në MYSQL i janë dhënë ID nga 1 deri tek 4.

3.6 Menaxhimi i Elasticitetit në Aplikacionet Cloud

Menaxhimi i elasticitetit, i njohur gjithashtu si shkallëzim sipas kërkesës të aplikacioneve është një veçori e rëndësishme në Cloud. Ne e kemi përcaktuar elasticitetin e nivelit të aplikimit si aftësia për të shkallëzuar burimet duke i rritur/e zvogëluar në mënyrë që të përmbushen kërkesat e performancës së aplikacionit siç është koha e kërkuar për përgjigje. Ekzistojnë tre role në menaxhimin e elasticitetit të aplikacionit në Cloud të cilët po i paraqesim në figurën e më poshtme.

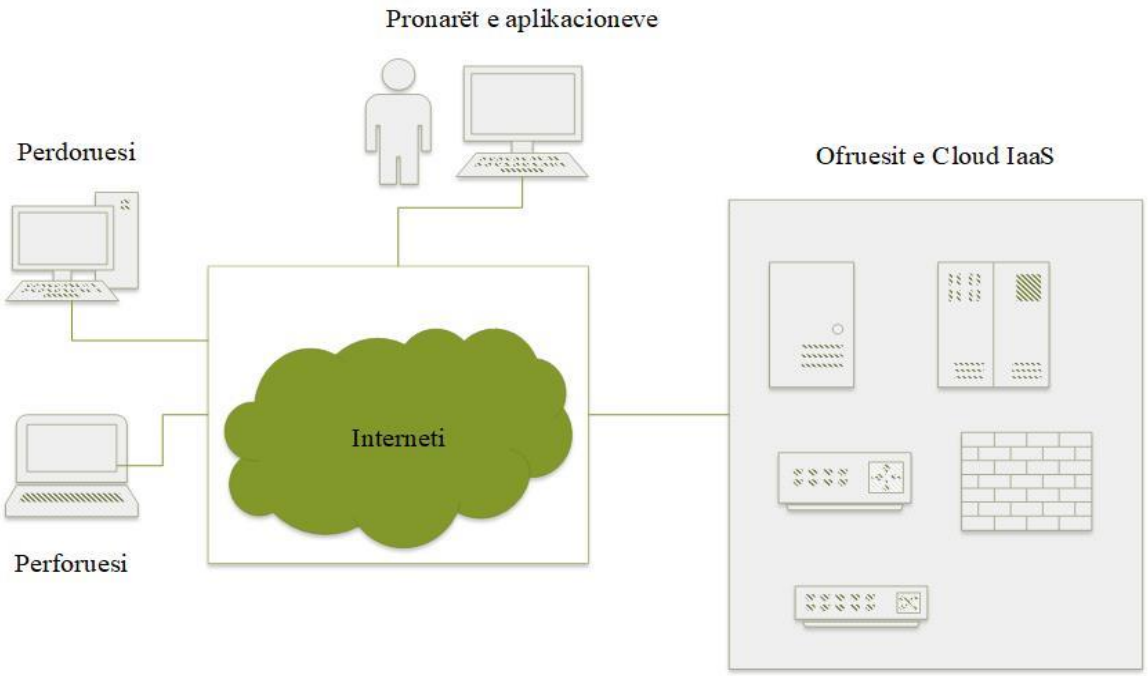


Figura 3 7 Paraqitet shembulli se si menaxhohen Elasticiteti në Cloud

ÅOfruesit e Cloud IaaS

Këta ofrues janë pronarë ose rishitës të infrastrukturës që furnizojnë me burime infrastrukturore ku përfshihet edhe burimet si Makinat Virtuale (VM); burimet e ruajtjes si hapësira e ruajtëse në internet; dhe burimet e rrjetit të tilla si adresat IP dhe gjerësia e rrjetit.

ÅPronarët e aplikacioneve

Pronarët e aplikacioneve, ose ofruesit e shërbimeve të aplikacioneve, janë konsumatorë të IaaS Cloud të cilët blejnë ose marrin me qira burime për të vendosur aplikacionet e tyre dhe ofrojnë këtë shërbime aplikacioni për palë të treta përmes internetit. Me fjalë të tjera, pronarët e aplikacioneve janë konsumatorët e IaaS Cloud dhe, në të njëjtën kohë, ofruesit e shërbimeve të tyre.

ÅPërdoruesit fundorë

Përdoruesit fundorë (end-user) të aplikacioneve, ose klientët e aplikacioneve përbëjnë konsumotorët e drejtpërdrejtë shërbime. Për shembull, përdoruesit përfundimtarë mund të shfletojnë faqet e internetit, të kërkojnë dhe të rendisin artikuj dhe të bëjnë porosi.

Targeti kryesor i elasticitetit në nivel aplikimi është të ndihmojë pronarët e aplikacioneve të bëjnë shkallëzimin e aplikacioneve të tyre për të arritur koston më të vogël të mundshme të vendosjes së aplikacionit si dhe të plotësojnë pritshmëritë e përdoruesëve. Kuptohet që bëhet fjalë për koston që gjenerohet nga përdorimi i burimeve infrastrukturore në Cloud duke ruajtur cilësinë e shërbimeve (QoS) siç mund të jetë koha e përgjigjes për përdoruesit e tyre përfundimtar.

Menaxhimi i elasticitetit në nivelin e aplikimit ka kuptime të trefishta:

ÅSë pari, nevojitet një shkallëzim dinamik për t'iu përgjigjur kërkesave të ndryshueshme të përdoruesëve përfundimtarë. Pra një shkallëzim automatik i aplikacioneve për të arritur një përformancë të pranueshme pa pasur nevojë të mbyllet për mirëmbajtje shërbimi i ofruar.

ÅSë dyti, shkallëzimi duhet të kryhet për të ruajtur QoS-në e një aplikacioni nën ndryshimin e kërkesave për burimet e serverave në shtresa të ndryshme të aplikacionit. Një pikë interesante këtu është se sasia e burimeve të konsumuara nga këta serverë varet nga sjellja e aplikacionit të vetë përdoruesit përfundimtar. Për shembull, nëse kërkimi i faqes në internet dhe kërkimi i produktit janë veprimet kryesore të përdoruesit përfundimtar për një periudhë kohe, serverat e internetit Apache dhe serverat e aplikacioneve Ameoba mund të stresohen dhe burimet e tyre të shterojnë. Nga ana tjetër, komponenti i bazës së të dhënave do të vazhdojnë të performojë mirë. Kur një aplikacion shkallëzohet lart ose

poshtë është thelbësore që të zbulohen pengesat e vërteta që mund të shkaktohen në cilindo, ose të gjithë shtresat ose serverat.

Menaxhimi klasik i sistemeve në Cloud është i bazuar në rregulla statikë, që me ezaurimin e burimeve realizohet dhe shkallëzueshmëria. Në menaxhimin e kohëve të fundit i vihet shumë theks manaxhimit të SLA/SLO si parametrat kryesorë që ofrojnë mundësinë e plotësimit të nevojave dhe kërkesës së përdorueseve në përgjithësi.

Është investuar së tepërmi në thjeshtimin e proceseve për marrjen apo heqjen e burimeve në përdorim për çdo përdorues sipas nevojës. Kjo është arritur duke zhvilluar modele analitike për vlerësimin e kapaciteteve që nuk kërkojnë shumë kohë për nxjerrjen e rezultateve që ndihmojnë në vendimmarrje, nga ku kjo vendimmarrje do të ketë efekt të drejtëpërdrejtë mbi kostot.

Nëse do të kemi një aplikim web për blerje produktesh atëherë do të na duhej të matnim kohën e përgjigjes për përdoruesit që vendosin të blejnë diçka në atë portal. Nëse do të kishim një rritje të kërkesave të shprehur në numër kërkesash atëherë do të kishim një rritje të kohës së përgjigjes. Ofruesi i shërbimit do të duhej të merrte burime shtesë në përdorim (të paguante më shumë). Me uljen e kërkesave të përdorueseve në aplikim do të ulet dhe koha e përgjigjes dhe mund të bëhet një vlerësim mbi gjendjen e burimeve informatike në dispozicion. Nëse koha e përgjigjes do të ishte shumë e ulët me heqjen e burimeve në përdorim koha e përgjigjes do ti afrohej limiteve të përcaktuara nga por pa e tejkaluar atë. Në këtë mënyrë me heqjen e burimeve në përdorim atëherë do të uleshin dhe kosto e ofrimit të aplikimit në total, pra kostot operacionale. Ky lloj menaxhimi fleksibël është një menaxhim elastik i aplikimeve/sistemeve në Cloud dhe ky menaxhim përfshin marrjen apo lëshimin e burimeve në përdorim në kohë reale duke e zgjeruar apo zvogëluar infrastrukturën në funksion të aplikimit apo bazës së të dhënave. Duhet të kuptojmë që menaxhimi elastik i burimeve në vetvete nuk ndryshon asgjë për nga vlera e marre nga baza e të dhnave/aplikimi pra rezultatet janë të njëjta. Qëllimi i përdorimit të shallzueshmërisë/elasticitetit është mbajtja e performancës brenda parametrave të paravendosur nëpërmjet SLA/SLO si element bazë për monitorimin e përmbushjes së

cilësisë së një shërbimi të kontraktuar midis sy palësh: ofruesi i shërbimit dhe përdoruesi i shërbimit.

Menaxhimi i kostove në skenarët e rregullimit automatik të shkallëzueshmërisë mund të bëhet kompleks në rast se dhe çmimet e marrjes në përdorim të burimeve të disponueshme bëhen variabël nga ofruesit e Cloud ku këto varen nga raporti kërkesë/ofertë.

3.7 Tipet e Teknikave të Elasticitetit të IaaS

Ka metoda dhe teknika të ndryshme që përdoren për të arritur shkallëzueshmëri për kapacitet/ burimet në mjedisin e IaaS Cloud [46]. Teknika kryesore klasifikohen në dy kategori për modelin IaaS: Reaktive dhe Proaktive.

Me teknikën Proaktive, vendimet për shkallëzueshmërinë bazohen kryesisht në modelin parashikues për të përcaktuar se si dhe kur do të ndodhë shkallëzueshmëria në IaaS Cloud [46]. Kjo qasje lejon shkallëzueshmëri automatike dhe të vetë adaptuar për ndryshimin e burimeve dhe kapacitete.

Në teknikën reaktive, vendimi për shkallëzueshmërinë bazohet tek ngjarjet ose kombinimi i ngjarjeve si p.sh. ndryshimi i sistemit, matja e burimeve, ofruesi i aplikimit. Teknika me e rëndësishme për këtë teknikë është elasticiteti i bazuar në rregulla (Rule Based). Ka shumë studime që merren me shkallëzueshmërinë me bazë rregullash [47]. Kjo është teknika më e përdorur sepse jep një mënyrë shumë të prekshme për të matur dhe për të vendosur kur dhe se si burimet e Cloud duhet të shkallëzohen.

Punimi i këtij disertacioni do të bazohet tek teknika e bazuara në rregulla i kombinuar me modelimin e sistemeve.

Në lidhje me strukturën dhe tipin e shkallëzueshmërisë së bazuar në rregulla, rregullat bazohen në dy pjesë që janë kushtet dhe veprimi. Pra duke vlerësuar kushtet merret një vendim për të nxitur një veprim elastik.

KAPITULLI IV

Punime të ngjashme

Në këtë kapitull do të paraqes punimeve shkencorë të kësaj fushe që më kanë shërbyer për thellimin e njohurive në kërkimin tim si dhe që kanë patur vend për optimizim ose zhvillim të mëtejshëm. Punimet të cilat më kanë shërbyer do ti ndaja në kate grupime: 1) punime që janë fokusuar tek shkallëzueshmëria në Cloud 2) punime që kanë të bëjnë me konceptet dhe rëndësinë e plaftformave të monitorimit dhe saktësinë e vlerave të monitorimit.

4.1 Shkallëzueshmëria dhe elasticiteti

Elasticiteti si një rast i vecantë i një sjellje të pavarur të sistemeve kompjuterike në Cloud dhe si mundësi prezente në të, është studiuar në shumë punime me varietete këndvështrimesh nga një komunitet i gjërë kërkuesish. Sipas disponueshmëria [49] është një nga cilësitë kryesore të Cloud ku burimet vihen në dispozicion sipas nevojës dhe hiqen nga përdorimi përsëri sipas nevojës. Në këtë punim që është një nga themelorët që më ka shërbyer për ngritjen e bazës teorike në lidhje me menaxhimin e elasticitetit. Ajo ofron një rishikim të rifreskuar të gjithë teknikave aktuale dhe përqasjeve ndaj elasticitetit në Cloud. Qëllimi i këtij kërkimi është klasifikimi në mënyrë të grumbulluar të këtyre përqasjeve. Kategoritë kryesore që kateorizojnë teknikat e elasticitetit janë:

- Fusha
- Qëllimi
- Vendimmarrja
- Veprimi Elastik
- Ofruesi
- Vlerësimi

Në këtë punim janë përmendur rreth 38 propozime në lidhje me elasticiteti në Cloud ku secila prej tyre paraqet veçantitë dhe përparësitë e veta ku qëllimi mbetet i njëjtë, një menaxhim sa më efektiv dhe që përballon kërkesat me performancë të kënaqëshme. Në to propozohen qasje të ndryshme, si ajo specifikimi i rregullave të paqarta, ku njohuritë e palëve të interesit janë analizuar tashmë dhe të ruajtura, duke lejuar përdoruesin të përcaktojë pragjet e nivelit të lartë, të cilat janë automatikisht të regjistruar në mënyrë rregullt bazuar në pragje konkrete. Sipas [49] qasje të tjera propozojnë transformimin e SLA-s dhe SLO-s në rregulla të bazuara në prag, duke përdorur rendet dhe pragjet e specifikimit të SLA dhe SLO dhe rregullat për zgjidhjen e konflikteve. Në në rastin e parashikimit të ngarkesës në të ardhmen, ka qasje të shumta, të cilat merren me pasaktësitë e parashikimit.

Sipas [50] paraqitet një model me bazë parashikueshmërie e përshtatshme për aplikime web. Qasja e propozuar ofron organizim automatik dhe shkallëzim proaktiv të shumëfishtë i njëkohshëm për aplikacione Web në një Infrastrukturë të caktuar të ndarë midis shumë klientësh. Ajo, monitoron dhe përdor burimet përdorimin e matjeve dhe nuk kërkon një model performance të aplikacioneve ose dinamikës së infrastrukturës. Ambienti Cloud na lejon të ndajmë makinën virtuale (VM) burimet midis aplikacioneve të vendosura, duke zvogëluar numrin që kërkohet për VM. Qasja demonstrohet në një prototip implementimi që është vendosur në Amazon EC2 Cloud. Ky lloj algoritmi sipas rezultateve ofron një menaxhim të mirë të QoS në raport me kohën e përgjigjes dhe burimeve në Cloud të përdorura.

Në [51] propozohet një model i profilizuar për shkallëzueshmërinë në kohë reale. Në këtë punim, profilet përdoren për të kapur njohuritë ñeksperteò për shkallëzimin e llojeve të ndryshme të aplikacioneve. Qasja e bazuar në profile, automatizon vendosjen dhe shkallëzimin e aplikimet në Cloud. Shkallëzimi në kohë reale arrihet pa u lidhur me infrastrukturën specifike të ofruesit të Cloud. Një rast real është përdorur për të demonstruar procesin dhe mënyrën e realizimit të përqasjes të implementimit të shallëzueshmërisë me referencë profilin.

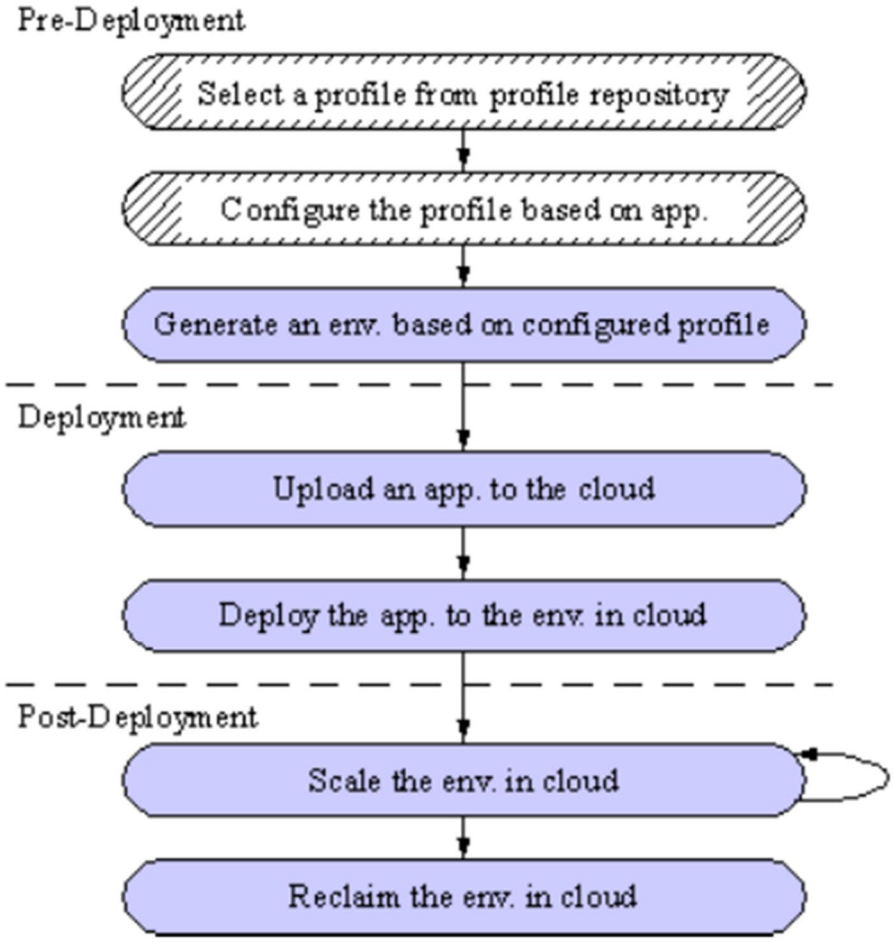


Figura 4 1 Implementimi i Shkallwzueshmwrisw sipas [51]

Ky model është një qasje që kërkon profilizim sipas cdo aplikimi. Është i imeplentuar në virtualizim me bazë Xen. Problematika e këtij algoritmi është trajnimi dhe profilizimi i çdo aplikimi të mudshëm që mund të vendoset në Cloud.

Në [52] është implementuar vendosja në dispozicion e burimeve për implementimin e një përqasje vetë-adaptuese duke përdorur agjentë softuerikë. Punimi përfshin edhe formulën e Kthimit të Investimit (ROI) që merret me marrëdhëniet midis çmimeve për Infrastrukturën-si-një Shërbim (IaaS) e kontraktuar nga klienti dhe përdorimi efektiv e këtij shërbimi Rezultatet eksperimentale tregojnë një përmirësim të dukshëm kur përdoret vetë-konfigurimi me llogaritjen e bazuar në agjent modelimi në kontrast me vetë-konfigurimin bazuar në parashikimin- tion për ngarkesën e ardhshme të punës. Sipas eksperimenteve të

paraqitura në këtë punim kemi një mbingarkim infrastrukture shumë pranë limiteve, por demonstrohet qartë një menaxhim i kostove në mënyrën më të mirë të mundshme.

Sipas [53] që kontribuon në përshkrimin e një arkitekture që menaxhon në mënyrë dinamike sjelljen e aplikimeve të vendosura në Cloud në bazën e disa rregullave në nivel të lartë. Arkitektura është fleksibël me mundësi modifikimi të rregullave gjatë egzekutimit të aplikimeve. Ajo mund të menaxhojë dhe aplikime të vendosura në ofrues të ndryshëm të Cloud. Ky implementim është kryer me ndihmën e prototipit Clotho. Janë realizuar teste të ndryshme performance dhe krahasime me implementime të tjera.

Në [44] prezantohet **soCloud** një platformë e orientuar drejt shërbimeve për menaxhimin e elasticitetit dhe shkallëzueshmërisë, lëvizshmërisë midis Cloud-eve të ndryshme. Kjo platformë bazohet në standardin OASIS Service Component Architecture për të adresuar lëvizshmërinë e aplikimeve.

Në [55] analizohen të mirat e të pasurit një shkallëzueshmëri të integruar brenda Cloud për menaxhimin e saj në shtresa të ndryshme të Cloud Nga studimi jepen dhe disa rezultate eksperimentale si dhe jepen disa rregulla për menaxhimin e shkallëzueshmërisë dhe mbajtjen e performancës në nivele të kënaqëshme. Në përfundimet e përfshira në të thuhet që qenia e integruar e shkallëzueshmërisë në Cloud ka avantazhe ku është eksperimentuar dhe konkurrenca midsi shtresave ku nakinat virtuale të aplikimeve konkurrojnë me ato të bazave të dhënave për burime dhe performancë.

Në [56] është bërë një rishikim i gjithë literaturës, përcaktimeve dhe parametrave monitorues. Ky dokument më ka shërbyer më së shumti në përcaktimin KPI në algoritmin tim si dhe më ka ndihmuar për të ndërtuar një bazë njohurish mbi trajtime shkencore të bërë nga kërkues të ndryshëm. Objekti i këtij studimi janë rreth 418 studime.Nëpërmjet këtij dokumenti përcaktova dhe parametrat bazë për SLA/SLO në algoritmin tim.

Në [57] integrohen parimet e shpërndarjes së automatizuar me vendimmarrje të shpërndarë. Në kontrast nga varientet e tjera të prezantuar me vendimmarrje qendrore që është dhe opisoni tradicional i menaxhimit automatik të shpërndarjes së ngarkesës. Sistemi autonom i menaxhimit dhe shpërndarjes së ngarkesës është i bazuar në modelin MAPE-K i propozuar nga IBM [57] si në figure.

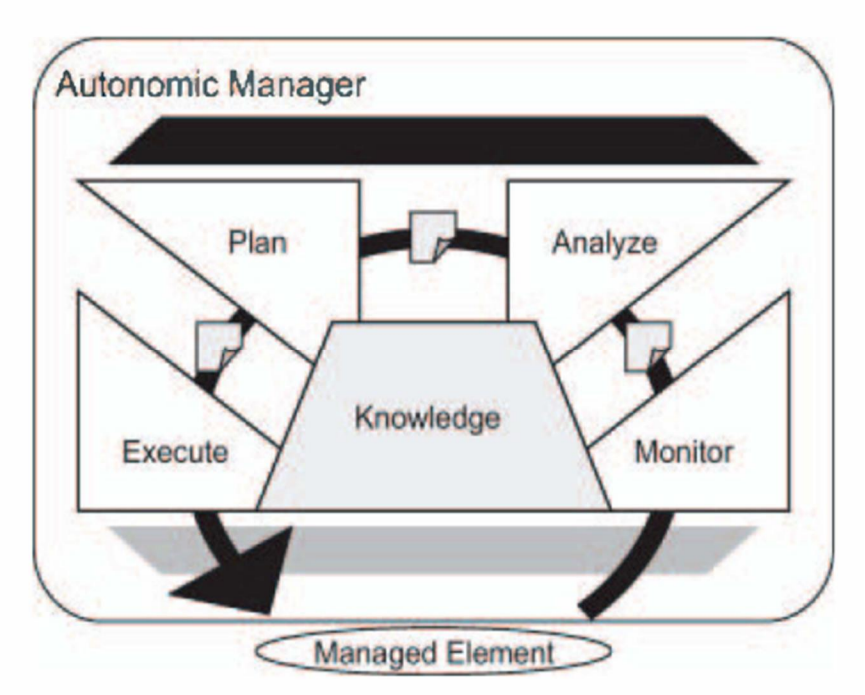


Figura 4 2 Menaxhimi autonom dhe shpërndarja e ngarkesës bazuar në modelin MAPE-K [57]

Në këtë punim është prezantuar një sistem që trajon shkallëzueshmërinë automatike në Cloud. Në të paraqiten problemet që hasen në ambiente komplekse të ambientit Cloud. Algoritmi i prezantuar është një zgjidhje e përgjithëshme që ndan kërkesat e sjella në shtresën e rrjetit në një model për të hapur ngarkesën në makina të tjera virtuale. Në këtë punim janë jashtë fushës së studimit ofrimi i bruimeve të instancave me aplikime Web ose fizike

Në [58] trajtohet migirmi i makinave virtuale bazuar sipas kërkesave të klientëve. Ky punim bazohet në trafikun dhe shpërndarjen gjeografike të dhomave të serverëve. Përveç analizimit të trafiku algoritmi analizon dhe kërkesat se ku kërkesat e përdoruesëve duhet të vendosen. Kërkesat vendosen tek dhoma e serverëve më e afërt dhe më me pak trafik. Arkitektura e propozuar në këte punim është paraqitur në figurë.

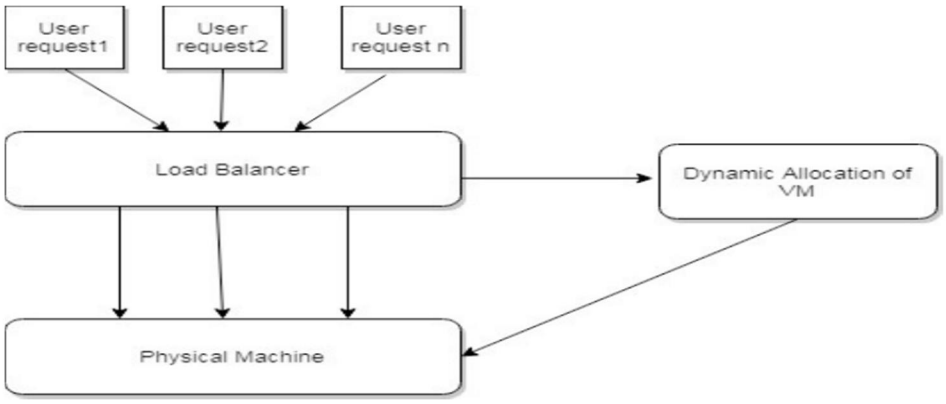


Figura 4 3 Arkitektura e propozuar sipas [58]

Qëllimi i këtij punimi është arritja e pritshmërive të përdoruesit e që shpeshherë shprehet edhe si SLA. Në eksperimentin e kryer shikohet që përgjigja e aplikimit përmirësohet me kalimin e kohës dhe kërkesat menaxhohen më mire. Ky lloj këndvështrimi është tepër interesant për sisteme të vendosura në ambiente të ndryshme gjeografike.

Në [59] propozohet një sistem (Kingfisher) i orientuar drejt kostove, që mundëson suport për elasticitetin në Cloud. Në këtë punim është bërë një krahasim dhe me kostot e shpenzuara me aplikimin e Kingfisher dhe atë aktual dhe përfitimi ka qenë me 24%. Me këtë algoritëm është realizuar një menaxhim efektiv i kostove.

4.2 Monitorimi i burimeve në Cloud

Pavarësisht se është një nga pikat të cilës të gjithë i referohemi në mënyrë rëndom si e mirëqenë apo e përcaktuar ku vlerat e ofruara prej saj janë lënda e parë e çdo lloj studimi monitorimi është procesi thelbësor i çdo sistemi përfshirë këtu edhe Cloud Computing.

Në paper [60] theksohet që vetë monitorimi i Cloud ka rëndësi për QoS. QoS është një term i përgjithshëm që ka të beje me performancën dhe me prioritetet në aplikime, për fushën që ne po studiojmë. Gjatë monitorimit në mënyrë dinamike regjistrohen të gjitha gjendjet e burimeve të virutalizuara si CPU, hapësira në disk, rrjeti etj. Teknikat e monitorimit ndihmojnë si ofruesin e Cloud ashtu edhe përdoruesin e saj të menaxhojnë burimet e tyre në mënyrë që aplikimet të egzekutohen me performancë të plotë. Monitorimi

i burimeve të Cloud ka rëndësi edhe për parandalimin e incidenteve në Cloud në vetevete ku një numër burimesh mund të dështojnë. Për këtë por dhe për arësye të tjera [60] e vë theksin në krijimin dhe propozimin e monitoruesëve dinamikë në Cloud Computing, kjo edhe për shkakun që Cloud Computing është vendi ku më së shumti dhe krejt normalisht ndodhin proceset e shkallëzueshmërisë dhe elasticitetit. Në [61] bëhet një analizë e gjerë mbi mjetet e monitorimit në Cloud Computing. Në të përcaktohet dhe taksonomia e monitorimit në Cloud Computing. Vetë Cloud duhet të ofrojë disa kushte të para caktuara nga standartet e NIST për përdoruesit. Të cilat i rendisim:

- Vetëshërbim sipas nevojës.
- Akses i gjerë dhe pa limit
- Grumbullimi burimesh dhe vënie në dispozicion
- Elasticitet të shpejtë
- Shërbim të matshëm

Për sa i përket shërbimit të matshëm ka mjete ose forma të paracaktuara për tëiu vënë ne dispozicion përdoruesit. Ofruesi i Cloud ofron mjetet dhe mënyrat për monitorimin e shërbimit por edhe për monitorimin me qëllim shfrytëzimin sa më shumë të Cloud për interes të shërbimeve që mund të ngremë në burimet e marra me qera (p.sh elasticiteti). Nga standardi i NIST për kërkesat ndaj Cloud dalin si rrjedhojë llogjike kërkesat për kornizat e monitroimit në Cloud. Në paper [63] procesi i monitorimit kalon në disa faza sipas figurës më poshtë.

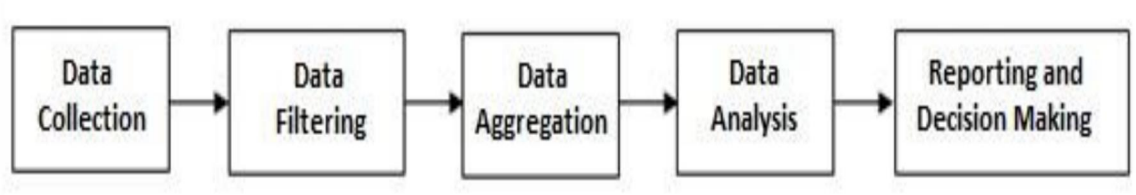


Figura 4 4 Fazat e Procesit të Monitorimit sipas [63]

Ku, në fazën e ñData Collectionò sistemi i monitorimit mbledh të dhëna të ndryshme si p.sh koha e procesimit, shpejtësia e procesimit, përdorimi i memories, konsumi i energjisë dhe përdorimi i saj. Të dhënat mund të mblidhen në mënyrë te centralizuar ose të decentralizuar.

Filtrimi i të dhënave është procesi ku merren të dhënat e nevojshme dhe që janë të rëndësishme për monitorimin apo qëllimin që do të përdoren këto të dhëna. Mënyrat e filtrimit mund të jenë të bazuara në kohë, periudhë të caktuar, në bazë të përmbajtjes dhe në bazë të të dhënave që kapërcejnë pragun.

Grumbullimi i të dhënave është një proces në të cilin informacioni i mbledhur dhe i shprehur në një formë të përmbledhur dhe statistikore.

Në fazën e analizës së të dhënave është pjesa ku përdoruesi i Cloud ka mundësi të shohë të dhënat e grumbulluara.

Raportimi dhe vendimmarrja është procesi përfundimtar ku gjenerohen raportet e përcaktuar, ku këto raporte do të shërbenjnë për marrjen e vendimeve në implementimin e kontrolleve të caktuar.

Në paper [64] është bërë dallimi midis fazave të fillesave të monitorimit në Cloud, ku cdo gjë monitorohej në mënyrë statike dhe pa qëllim në vetvete ndërsa nuk ishin të përshtatura për monitorimin e performancës që është diçka e paparashikueshme në vetvete dhe që cënimi i tyre prek SLA/SLO. Gjithashtu ndryshimet në gjerësi brezi, në disponueshmëri burimesh, dhe kapacitete të ndryshueshme dhe që mund të mos jenë në përputhje me kërkesën janë të gjithë faktorë që ndikojnë në SLA. SLA është një nga pjesët më të rëndësishme të një kontrate dypalëshe dhe është pjesë ligjore brenda çdo kontrate të dakordësuar. Që SLA të jetë bindëse një sistem raportimi i krahasueshëm nga ofruesi dhe klienti dhe që ka raporte të përacktuara në bazë të SLA e bën mardhënien kontraktuale transparente. Vetëm nëpërmjet monitorimit të performancës mund të ngrihen KPI (Treguesit Kyc të performancës) për patformat ashtu dhe për aplikimet. Për eliminimin e këtyre pasigurive janë zhvilluar dhe perfeksionuar teknikat e monitorimit që mund të ndihmojnë ofruesit e Cloud dhe zotëruesit e aplikimeve për patjen sa më transparente të mënyrës se si Cloud he aplikimet e vendosura në to, performojnë. Pjesa më e madhe e ofruesëve aktualë të Cloud monitorojnë makinën virtuale pa u futur në detaje mbi aplikimet e vendosura brenda tyre dhe vetëm një pjesë e vogël e mundësojnë këtë. Pavarësisht kësaj asnjë nga mjetet e monitorimit nuk ka mundësi për vendosjen e rregullave apo politikave të respektimit të limiteve të QoS. Monitorimi i QoS për aplikimet është një

fushë që ka studime të shumta por nga vetë natyra e ndryshme e vetë aplikimeve (si biznesi, shkencore etj). Përgjithësimi i një kornize dhe mekanizmi kontrolli është një nga sfidat e momentit në këtë drejtim. Sipas [65] monitorimi i Cloud është është një aktivitet i rëndësishëm e kjo sidomos për QoS. Është monitorimi ai që krijon mundësinë e krijimit të mekanizmave për parandalimin dhe normalizimin e situatave të paparashikuara. Cloud Commputing Në të i jepet rëndësi monitorimit midis shtresave për të kuptuar dhe realizuar ndarjen midis shtresave në Cloud dhe për të optimizuar shtresën që ka rënie performance. Përdoruesit nuk kanë të drejta për aksesimin e shtresave të poshtme ndërsa ofruesit e Cloud nuk kanë akses në shtresat e larta. Si pasojë të dyja palët marrin vendime për pjesën për të cilën ata kanë të dhëna nga monitorimi.

Në [66] autorët kanë kryer një sondazh në lidhje me monitorimin në Cloud Computing. Ata theksojnë rëndësinë e montiroimit të Cloud dhe japin konceptin se çdo aktivitet dhe veçori e ofruar në Cloud ka nevojë të monitorohet.

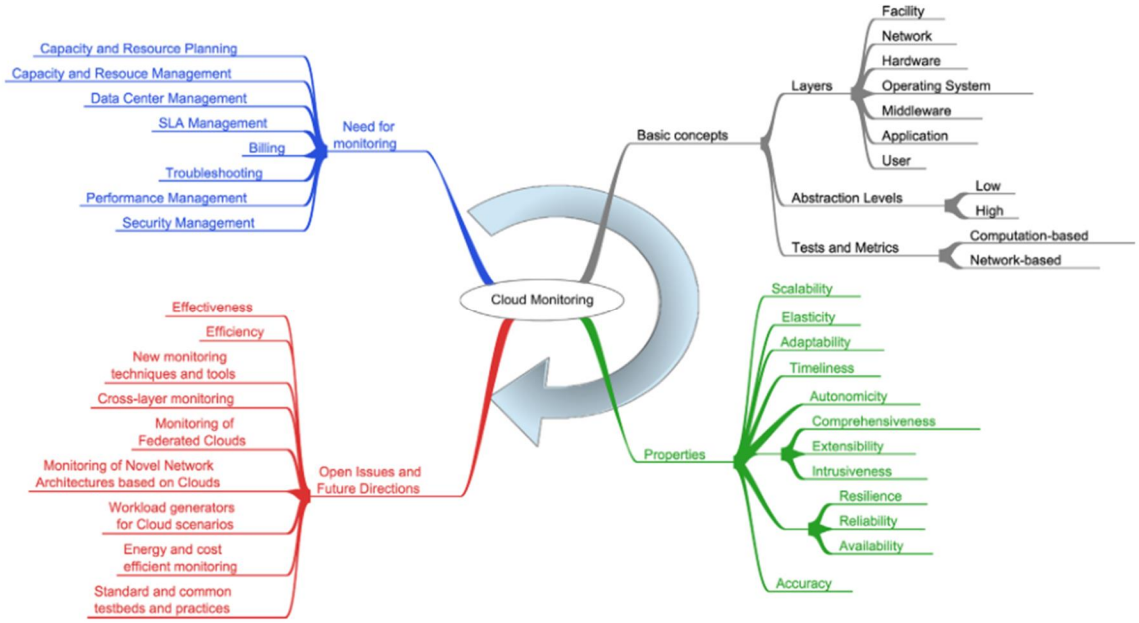


Figura 4 5 Monitorimi i Cloud (motivations, properties, basic concepts, open issues and future directions) [66]

Aktivitetet që kanë nevojë për monitorim në përgjithësi por që nuk kufizohen vetëm në to janë:

- Planifikimi i burimeve dhe kapacitetit

- Menaxhimi i kapaciteteve dhe burimeve
- Menaxhmi i ñData-centerò
- Menaxhimi i SLA
- Faturimi
- Zgjidhja e problemeve
- Menaxhimi i performancës
- Menaxhimi i Sigurisë

Sipas [67] fokusohet në monitorimin në Cloud hibrid që në parim është i ngjajshëm me të gjithë monitorimet e mirëfillta të Cloud. Në të jepen dhe parametrat e monitorueshëm dhe qëllimi për të cilin mund të përdoren këto parametra.

Layer	***Type***	***Key Metrics***
Virtual Infrastructure	Components	number of CPU, memory size, hosting space, consumed time, cost per instance
Network	Throughput	bandwidth
	Availability	data lost rate, data error rate
	Efficiency	Response time (average/ maximum),
Application/ Service	Service response time	Transfer time, max ,min , average service time
	Service cost	Consumed time, billing

Tabela 4 1 Metrics for client side monitoring

Në [68] prezantohet arkitektura e cila përbëhet nga një Sistem me Shumë Agjentë që përdoret nga një Cloud për të ndihmuar në përcaktimin e burimeve më të mira dhe për të krijuar metodën e negocimit midis ofruesit të Cloud dhe përdoruesit për të shfrytëzuar aftësinë e plotë të platformës së Cloud. Gjithashtu kjo arkitekturë është projektuar kështu që mund të monitorojë aplikimet e përdoruesit ndërsa ato janë në fnksion. Në të vihet theksi për rëndësinë dhe vlerën e shtuar të sistemit me shumë agjentë ku monitorimi dhe brenda makinës virtuale në IaaS është i arritshëm plotësisht.

Një nga veçoritë e krahasimore sipas Fig. 4.5 është dhe besueshmëria (Reliability). Besuashmëria është veçoria kryesore për të cilën varet dhe saktësia dhe egzekutimi i

kërkesave sipas konfigurimit ose dëshirës për tërheqjen e të dhënave të monitorimit kundrejt makinave virutale apo LB [99]. Sipas [99] përveç karakteristikave kryesore të krahasimit të sistemeve të monitorimit të Cloud është bërë një krahasim midis produkteve të ofruara nga operatorët Cloud si dhe me programet me burim të hapur që përdoren për monitorim. Sipas krahasimeve sistemet e integruara në Cloud plotësojnë shumicën e karakterisitikave të një sistemi monitorimi në Cloud. Ndërsa pjesa tjetër e programeve me burim të hapur një pjesë e tyre nuk e plotësojnë kushtin e besueshmërisë. Për më tepër vendosja e këtyre sistemeve me qëllim monitorimin e makinave virutale dhe LB kërkon vendosjen e një makine virtuale. Sistemet e monitorimit kërkojnë parametra të lartë harduare-ik (fizik apo virtuale), për arësye të monitorimit të shumë burimeve në të njëjtën kohë. Pasja e një serveri të tillë makinë virtuale ose fizike do të kishte kosto shtesë për menaxhimin, monitorimin dhe evidentimin e problemeve në sistemet Cloud.

4.3 Përfundime

Këto punime ishin tepër të vlefshme për ngritjen e dhe plotësimin e njohurive të mia teorike dhe praktike për sa i përket Cloud, elasticitetit, shkallëzueshmërisë dhe monitorimit.

Për më tepër, nga studimet që prezantova në ketë seksion ky punim do të bazohet edhe tek modeli i platformës të studimit [59] për përdor profilizimin e makinave virtuale në disa nivele të përshtatur me madhësinë ose parametra e makinave virtuale të përcaktuara. Përqasja në këtë punim shfrytëzon replikimin dhe migrimin duke vënë në dispozicion kapacitetet dinamike dhe duke përdorur një program linear për llogaritjen e kostos. Ky model i paraqitur konsiderohet i përshtatshëm për të arritur qëllimin e këtij punimi që minizimin e burimeve të përdorura. Gjithsesi, studimi që paraqes ndryshon në lidhje me prototipin e përdorur. Në rastin studimin [59] përdoret OpenNebula si prototip ndërsa, ky studim është fokusuar tek Eucalyptus.

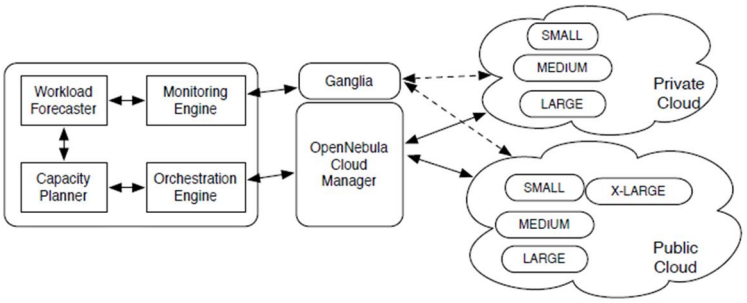


Figura 4 6 Arkitektura e propozuar nw studimin [59]

Siç e shohim dhe nga arkitektura e propozuar tek [59] pjesa e monitorimit është e përfshirë në të si element përbërës.

Në Eucalyptus ka monitorues të përfshirë brenda Cloud dhe që monitorojnë të gjitha instancat në detaj si dhe në parametrat e dëshiruar duke ofruar dhe mundësinë e ngritjes së alarmeve. Ajo që do të shohim ne përgjatë eksperimentit është edhe shpejtësia e reagimit të algoritmit në platformë.

KAPITULLI V

Analiza e problemit dhe propozimi

5.1. Hyrje

Në këtë kapitull do të analizojmë problemin pyetjen e ngritur dhe do të ofrojmë një zgjidhje llogjike si dhe implementimin në një ambient test.

5.2. Analiza e problemit

Ofruesit e përdorin modelin IaaS si bazë për tu mundësuar klientëve të tyre marrjen me qira të burimeve përpunues dhe hapësirës në Cloud sipas nevojës dhe në bazë përdorimi duke ofruar kështu modelin e tarifimit, pagesës sipas përdorimit. Ofruesit e Cloud rrisin fitimet në këtë mënyrë duke maksimizuar përdorimin burimeve në infrastrukturës fizike minimale.

Çdo kontratë me një ofrues Cloud ka një seksion të veçante për SLA (Service Level Agreement). Në të gjitha SLA ka pjesë ku nga ofruesi garantohen një sërë parametrash të matshëm nga përdoruesi që merr me qira burimet në Cloud. Por një parametër që nuk është i përfshirë nga ofruesit e Cloud në përgjithësi, në këtë seksion nuk është koha e përgjigjes e parë nga ana e përdoruesit. Kjo ndonjëherë edhe me të drejtë. Marrësi me qira i Cloud mund të implementojë edhe konfigurime të gabuara në aplikimet e ngritura prej tij të cilat mund të afektojnë eksperiencën e përdoruesit me aplikimin. Gjithsesi përdoruesi fundor gjithmonë do të kërkojë kohën e përgjigjes nga sistemi të përfshirë në termat SLA të vendosur ne kontratë mes tij dhe ofruesit të aplikimit (marrësi me qira i Cloud). Ofrimi i garancisë në vetvete për kohën e përgjigjes ka dy sfida në vetvete:

a. Trafiku drejt faqeve të Web është dinamik dhe i vështirë të parashikohet me saktësi

b. Në aplikimet shumë shtresore mund të ndodhin vonesa në në nga shtresat që mund të krijojnë në mënyrë indirekte vonesa në përgjigjen ndaj përdorueseve dhe këto lloj vonesash janë të vështira për tu identifikuar dhe për tu zgjidhur.

Gjithashtu është e vështirë vendosja e burimeve në mënyrë manuale për përballimin e ngarkesës. Që një ofrues Cloud të garantojë një kohë përgjigje maksimale do të duhet të ketë mekanizma të automatizuar në dispozicion për evidentimin e bllokimeve, raportimin e tyre si dhe marrjen e masave për zgjidhjen e tyre.

Një aspekt tjetër ku ofruesit e Cloud konkurrojnë me njëri tjetrin është dhe ai i tarifimit. Që ata të jenë sa më konkurrent në treg do të duhet të jenë sa më transparent dhe të ofrojnë mundësinë që përdorimi dinamik i burimeve të ofruara prej tyre, ti ofrojnë mundësinë përdorueseve që të përdorin pa asnjë kufizim të gjithë gamën e produkteve nga më i liri tek më i shtrenjti në mënyrë që ata të ofrojnë të njëjtën performancë duke menaxhar kostot e tyre në mënyrën më të mirë të mundshme dhe sipas dëshirës.

5.3 Parakushte

Për të patur një menaxhim sa më efektiv të saj duhet të investohet shumë në monitorim dhe saktësinë e monitorimit. Vetëm një monitorim efektiv dhe i parametrave të duhur krijon mundësinë e aplikimit sa me optimal të algoritmit të propozuar për menaxhimin e sistemit tonë. Këto parametra mund të jenë ngarkesa e CPU, memories, përdorimi i rrjetit (bandwidth). Ajo që do të implementojmë ne këtë temë ka te bëjë me një menaxhim automatik të burimeve kompjuterike në Cloud me qëllim që ato të jenë të shakallëzueshme dhe eficentë. Ky algoritëm do të ketë për qëllim:

1. Monitorimin e sistemit
2. Rregullimin e performancës
3. Menaxhimin e elasticitetit
4. Menaxhimin e ñLoad balancersò në varësi të ngarkesës

Ditët e sotme bazat e të dhënave janë pjesë integrale e çdo aplikimi dhe për këtë analizën do ta shtrijmë edhe në shtresat e tjera si të pandashme në këtë proçes.

Se fundmi ky proçes do të duhet të garantojë SLA sipas kërkesave të kontratës si dhe SLO sipas termave operacionalë të vendosur nga marrësi në përdorimi i këtyre burimeve. Sipas ngarkesës apo rregullave të vendosur merren vendimet përkatëse për rritje apo ulje shkallëzueshmërie, në nivel aplikimi/baze të dhënash apo në nivel makine virtuale (VM).

Egzistojnë shumë algoritma që mundësojnë realizimin e shkallëzueshmërisë në Cloud por vetëm disa prej tyre janë të orientuar drejt kostove. Specifikat dhe varësia se cilët parametra nga makina virtuale zgjidhen për monitorim janë ato që bëjnë diferencën dhe në algoritmat që mundësojnë shkallëzueshmërinë në Cloud.

Algoritmi për menaxhimin e elasticitetit është edhe më kompleks nëse do të aplikjmë dhe rregullat e biznesit (p.sh menaxhimi i kostove bazuar në një listë çmimesh të dhëna paraprake). Monitorimi në këto raste do të kërkonte dhe përcaktimin e buxheteve (të ardhura, shpenzime) si dhe rezultatin përfundimtar nëse do të kishim humbje apo fitim në skenarin e propozuar.

5.4 Modelimi i shtresave në Cloud

Për të ruajtur një balancë midis zvogëlimit të kostos dhe Marrëveshjes së Nivelit të Shërbimit (SLA), IBM ka propozuar një model për të menaxhuar në mënyrë të pavarur mekanzimin e shkallëzimit automatik në formën MAPE (Monitor-Analyze- Plan-Excute) si model për referencë [70]. Modelimi i shtresës së Cloud për shkallëzueshmëri të shpejtë fokusohet më shumë në analiza sasiore dhe në modelin MAPE [71]. Elementët kryesorë në këtë implementim janë:

Komponenti i monitorimit që mbledh të dhëna nga sistemi si psh: CPU, memorie, hapësirë në disk. Monitorimi ekzekutohet cdo 5 sek. Të dhënat e elementëve të interesit do të mblidhen dhe do të përpunohen ngmodeli i performancës.

Komponenti i analizës. Faza e analizimit [72] është përgjegjëse për analizimin e të dhënave. Analiza do të shërbejë për marrjen e vendimeve nëse do të implementohet shkallëzueshmëri apo jo në bazë të të dhënave që janë përpunuar nga komponenti i monitorimit.

Planifikimi. Eshtë komponenti kryesor në Cloud. Është kjo shtresë që kujdeset për planifikimin e shkallëzueshmërisë me koston më të vogël duke dhënë një ndikim direkt dhe tek energjia e konsumuar. Njëkohësisht do të ulte ose do të ngrinte numrin e serverëve dhe burimeve sipas limiteve të paracaktuara.

Ekzekutimi. Në fazën e ekzekutimit janë Load Balancer-at që përballojnë ngarkesën duke krijuar numrin e nevojshëm të serverëve në infrastrukture. Sipas parametrave të parackatuar do të krijohen makinat virtuale në serverat fizikë.

Ku model aplikohet tek sistemet e aplikimeve web që e njohin gjëndjen e tyre dhe regojnë për ndryshimin e tij. Cikli i kontrollit ripërsërit vetveten në mënyrë të vazhdueshme në kohë. Cikli fillon me monitorimin, më pas kalon tek analiza dhe planifikimi (rule-based planner) [72]. Në fazën e fundit kemi ekzekutimin e vendimit që do të thotë ky mekanizëm dërgon kërkesën për të instaluar ose hequr një makinë virtuale sipas vendimit që është marrë gjatë fazës së planifikimit.

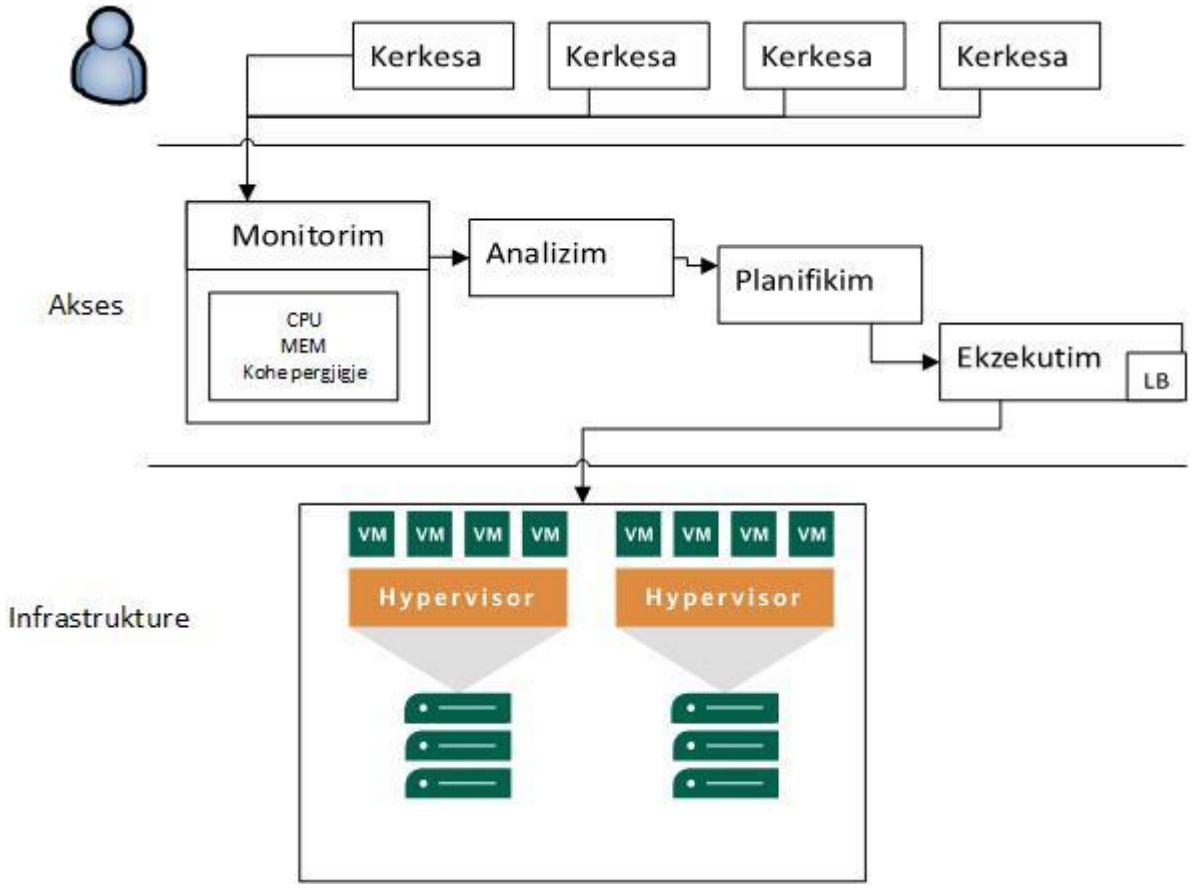


Figura 5 1 Skema efunksionimit të MAPE

5.5 Zgjidhja e propozuar

Algoritmi përdoret për marrjen apo lëshimin e burimeve në kohë reale për të mbajtur në nivelin e dëshiruar/kërkuar performancën e aplikimeve. Në se do të mundohemi të kuptojmë se sa do të ishte kosto totale e aplikimit do të na duhej thjesht të kishim një regjistër se sa herë është egzekutuar algoritmi si në ñscale-upò ashtu edhe në ñscale-downò si dhe kohën e qëndrimit të gjendjeve të ndryshme ku përfshihen numri dhe tipi i makinave virtuale ne egzekutim. Një regjistër i tillë egziston në ofruesit komercilaë të Cloud dhe shërben pikërisht për llogaritjen e kostove.

Këto kosto do të merren në mënyre empirike me qëllim ndarjen nga ofertuesit aktuale tregtarë duke patur ne fokus vetëm funksionimin e suksesshëm të algoritmit. Për sa i përket

kostove sic do të shohim edhe nga algoritmi do të realizohet shkallëzueshmëria sipas makinave me koston më të ulët.

Më poshtë do të japim disa varibala të cilët do të na shërbejnë për ta bërë më me kuptim algoritmin tonë:

Parameteri	**Përshkrimi**	**Burimi**
□(□)	kosto e një serveri	Ofruesi i Cloud
□□□	Koha e përgjigjes e kërkuar	SLA/SLO
□□	Buxheti në dispozicion	
□□	Numri i kërkesave në njësi kohe	Monitorim në kohë reale
□□	Koha e përgjigjes e monitoruar	

Tabela 5 1 Parametrat kryesorë tw Algoritmit

Një pjesë e këtyre vlerave do të jenë të paracaktuara dhe do të shërbejnë si elementë referencë në shkallëzim/elasticitet. Ato do të jenë kufijtë e sipërm ose të poshtëm për ñscale-upò ose ñscale-inò.

Menaxhim i elasticitetit në algoritëm do të bazohet në disa parametra të paracaktuar në raport me SLO/SLA dhe kostot për makinë virtuale. Parametrat e monitorimit të përcaktuar merren nga programe të integruara dhe i dërgohen algoritmit me qëllim analize dhe vendimmarrje. Parametrat e monitorimit janë mbi bazë makine virtuale dhe koha e përgjigjes si dhe ritmi i ardhjes së kërkesave mblidhen nga programe te ndryshëm në Cloud. Ky algoritëm do të aplikohet sa herë që një instance e re do të ekzekutohet. Alogritmi është si më poshtë:

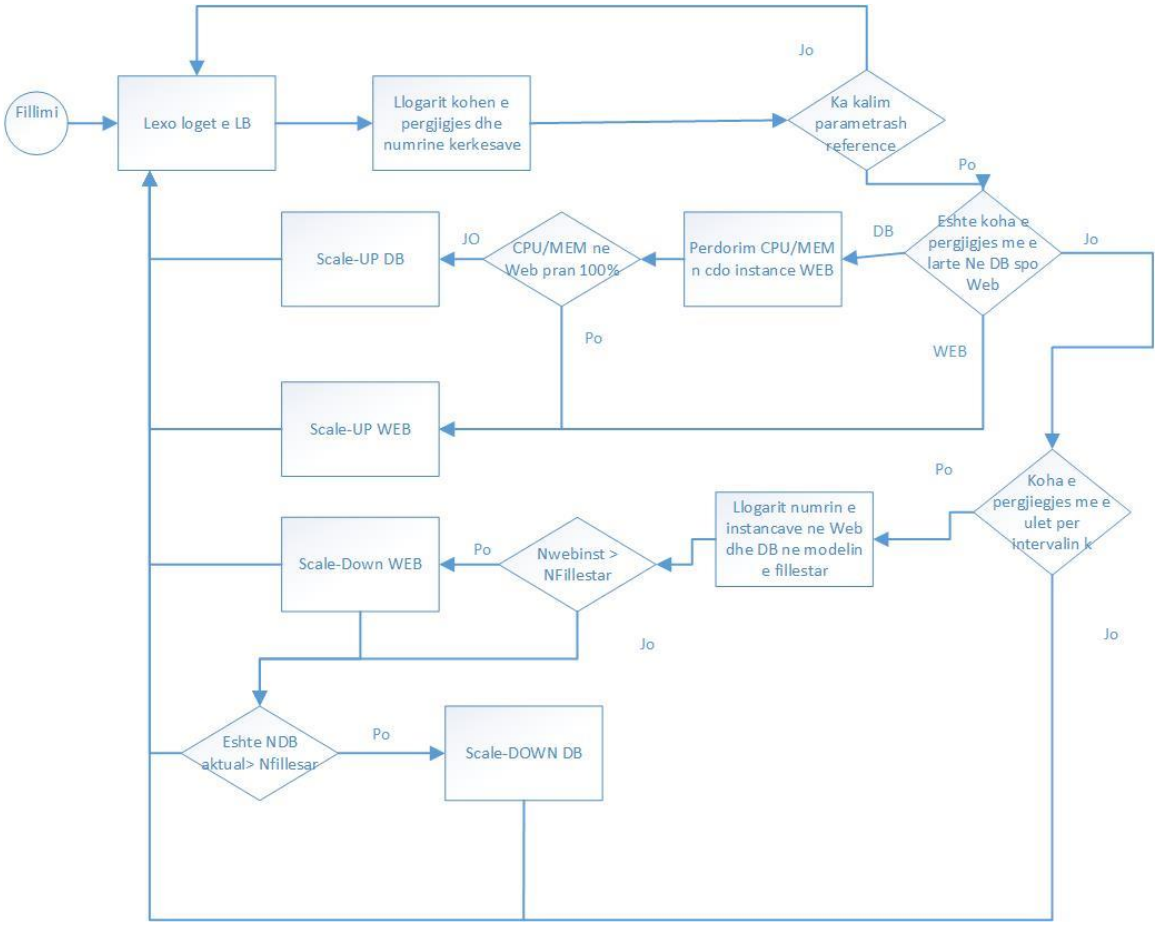


Figura 5 2 Algoritimi i propozuar

Algoritmi kontrollon në mënyrë periodike loget e Load Balancers sipas shtresave për kërkesat si dhe llogaritjen e kohës së përgjigjes. Kur kemi tejkalim të kohës së përgjigjes atëherë do të kontrollojmë dhe përdorimin e procesorit dhe memories në makinat virtuale ku kemi vendosur aplikimet web. Kontrolli i përdorimit të procesorit dhe memories në instancën ku qëndron aplikimi web bëhet për faktin nëse kemi faqe statike web duke u shërbyer ose nëse java servlet në vetvete ka rënie në performancë. Në rast se nuk kemi saturim atëherë atëherë kemi shkallëzim ne shtresën e bazave të të dhënave. Në rast të kundërt do të kemi shkallëzim të shtresës së Web. Algoritmi kontrollon në mënyrë periodike kohën e përgjigjes.

Në rast se do të kemi një kohë përgjigje më të vogël se koha e paracaktuar atëherë do të kontrollojmë numrin e serverëve aktualë në cdo shtresë me atë të serverëve në cdo

shtresë në kohën e inicimit të aplikimit.. Në rast se ky numër do të jetë më i madh se numri fillestar atëherë do të aplikojmë ñScale-Downò tek shtresa e cila ka servera më shumë se sa numri fillestar i serverëve.

Për blloqet ñScale-Up DBò, ñScale-Up Webò, ñScale-Down DBò, ñScale-Down Web ò do të japim algoritmat gjenerikë për ñScale-Upò dhe ñScale-Downò

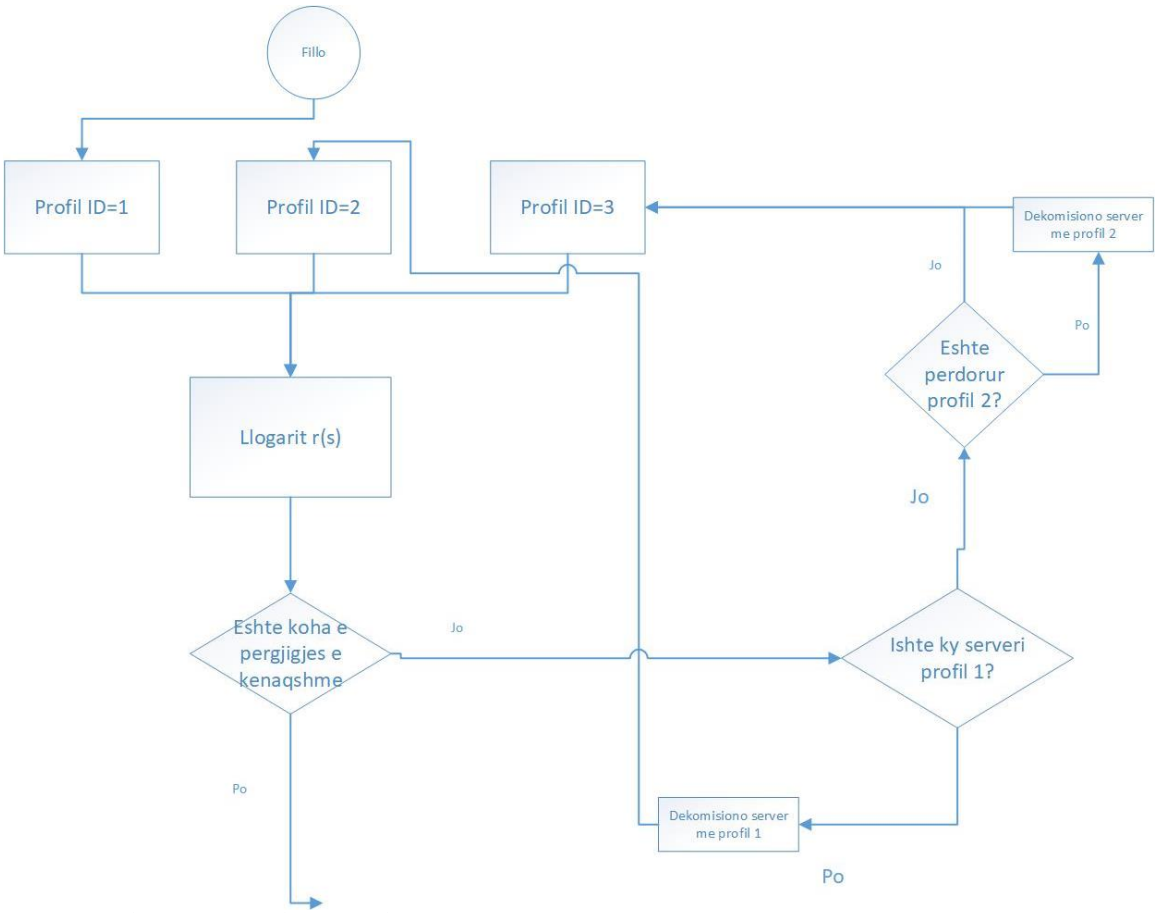


Figura 5 3 Algoritmi gjenerik për "Scale-Up"

Algoritmi ñScale-Upò ka për qëllim shtimin e serverëve të rinj me qëllim mbajtjen e performancës në nivelet e vendosur nga SLA/SLO duke tentuar të mbajë kostot në një nivel sa më të ulët. Sic shohim dhe nga algoritmi, kemi konceptuar 3 profile makinash virtuale të cilat do të përdoren gjatë procesit të ñScale-Upò. Në momentin që do të kemi marrjen e vendimit për ñScale-Upò në shtresën se ku do të bëhet ky proçes shtohet një makinë virtuale. Makina virtuale që shtohet ka parametrat më të ulët. Me shtimin e saj në shtrese

do të llogaritet koha e përgjigjes r(s). Në varësi të kohës së do të ndiqen hapat e mëtejshëm. Në rast se koha e përgjigjes është brenda SLA/SLO atëherë algoritmi vijon egzekutimin si në variantin kryesor. Nëse koha e përgjigjes nuk është e kënaqëshme, do të vlerësojmë nëse serveri i përdorur është ai me parametrat minimalë. Në rast se kjo është e vërtetë atëherë do të dekomisionohet serveri me parametra minimalë dhe do të realizohet live migration në server me parametra mesatarë. Në rast se koha e përgjigjes do të jetë e kënaqëshme atëherë do të vijohet me egzekutimin e algorimit kryesor. Në rast të kundërt ashtu siç shihet edhe ne figurën 5.3 do të kalohet nëpër hallkat analizuese dhe meqë është përdorur serveri me parametra mesatarë edhe ky do të dekomisionohet ose me mirë do kalohet në serverin me profil 3 që ka dhe parametrat maksimalë. Në rast se dhe me shkallësimin me serverin me parametrat maksimalë nuk do të arrihet koha e kënaqëshme atëherë do dtë shtohet një server i ri me parametra maksimalë në shtresën që ka nevojë për shkallëzueshmëri.

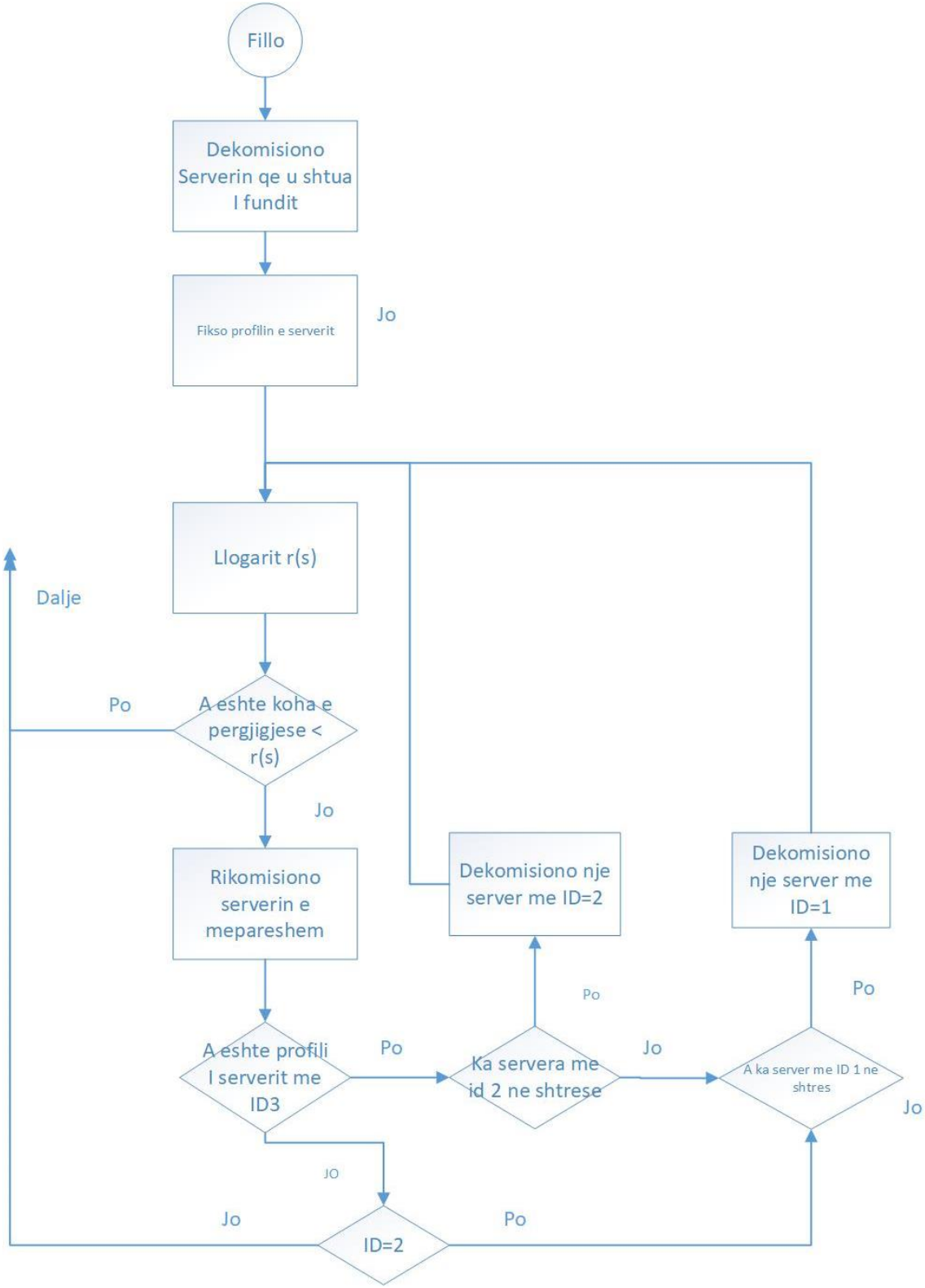


Figura 5 4 Algoritmi gjenerik për "Scale-Down"

Sic paraqitet në figurë algoritmi i ñScale-Downò, fillimisht algoritmi dekomisionon makinën virtuale që eshtë shtuar së fundmi dhe llogarit kohën e përgjigjes së aplikimit. Në rast se koha e përgjigjes pas heqjes se serverit të fundit të shtuar është më e lartë se ajo e pranuar atë herë do të rikomisionohet makina virtuale që sapo u dekomisionua. Nëse serveri i shtuar se fundmi ishte i me parametrat e profilit 3 atëherë do të shihet nëse në shtresë ka servera me parametra të profilit 2. Nëse ka servera të profilit 2 në shtresë atëherë do të kemi dekomisionim të një serveri me profil 2 (parametra fizike). Në rast se në shtresë nuk kemi server me profil 2 atëherë do të shohim nëse në shtresë ka servera të vendosur që kanë parametra fizike në përputhje me profilin 1. Në rast se kemi server të këtij tipi në shtresë atherë do bëjmë ñScale-Downò një nga serverat e profil 1. Në rast se ska atëherë do të egzekutohet dalja nga algoritmi dhe vijimi i algoritmit fillestar.

Fillimisht sipas modelimit të aplikimit vendosim një LoadBalancer, një webserver dhe një server baze të dhënash. Algoritmi punon në kushtet kur koha e përgjigjes K_{PM} është më e lartë K_{PD} dhe numri i serverëve mund të shtohet. Algoritmi zgjedh makinën virtuale që shton koston më të vogël në përdorim. Në rast se serveri i përzgjedhur nuk e ul K_{PM} atëherë algoritmi e dekomisionon dhe përzgjedh një server që ka parametra më të lartë. Ky veprim vazhdon për sa kohë arrihet qëllimi ku $K_{PM} <= K_{PD}$. Gjithsesi edhe shtimi i serverëve ka një limit të vendosur në algoritëm. Tejkalimi i këtij numri nuk lejohet. Ezaurimi i burimeve në Cloud kërkon ndërhyrje manuale bazat e të dhënave. Do të marrim të mirëqenë që algoritmi merr një kohë fikse për ekzekutim.

Algoritmi për ñScale-Downò ka për qëllim heqjen e serverëve me qëllim uljen e kostove por me kusht që parametrat e performancës së monitoruar të jenë brenda atyre të paracaktuar. Ky algoritëm egzekutohet deri në pikën kur nuk mund të hiqen më serverë redundant.

Krijimi një profili të paracaktuar VM do të marrë të mirëqenë nivelin maksimal të kërkesave që mund të suportojë VM dhe në bazë të diferencës së kërkesave ne njësi kohe

me ato që përballon sistemi do të merret dhe vendimi për shkallëzueshmëri dhe patje të një sistemi eficient. Me çmimin e referencës të vendosur do të llogaritet një kosto totale shfrytëzim infrastrukture Cloud.

5.6 Kriteret e vlerësimit

Të gjithë veprimet e shkallëzueshmërisë si ñScale-Upò ashtu edhe ñScale-Downò ndikojnë në kohën e përgjigjes si dhe në kostot totale të implementimit të aplikimit. Algoritmi ka për qëllim ngritjen e aplikimit dhe menaxhimin e kohës së përgjigjes me sa më pak kosto.

Teknikat e shkallëzimit ndahen në dy grupe të mëdha ato në bazë të një në bazë të shkallëzimit në bazë të rregullave dhe shkallëzimit në bazë të shtresave.

Në grupimin e parë këto algoritma përcaktojnë rregulla strikte për shkallëzueshmërinë. Këto tipe algoritmesh përdoren më së shumti nga ofruesit IaaS. Në këto raste në përgjithësi serveri ose serverat në bazë të parametrave të monitorimit për ñvotojnëò për shkallëzim atëherë shtohet një server. Koha e vlerësimit të shallzueshmërisë është vendosur 2 sek për ñScale-Upò dhe 1 sek për ñScale-Downò.

Kategoria e dytë e algoritmeve bazohen në modelimin e trafikut të rrjetit. Në këtë algoritëm trafiku në prurje krahasohet me skenarin më të keq të performancës (pra ndërtimi i një raporti të kërkesave me përgjigjen e matur) sipas profilizimit të makinës virtuale me parametra të caktuar. Një kohë e njëjtë është vendosur si për ñScale-Upò ashtu dhe ñScale-Downò. Me kapacitetin maksimal të mundshëm do të konsiderojmë situatën ku kemi një numër maksimal serverësh të implementuar dhe që përballojnë 130% të kërkesave ku patur pjesën e përgjijges në parametrat e SLA/SLO.

Të dy këto algoritma së bashku me algoritmin që propozojnë këtë punim i kanë kushtet fillestare të zbatimit të njëjta.

Për sa i përket numrit të instancave algoritmi i propozuar në këtë punim paraqet të njëjtin nivel performance me më pak servera. Ky rezultat del edhe më i qartë në rast se eksperimenti do të bëhet shumë shtresor. Algortimi nuk vepron në mënyrë statike por duke

parë vonesat në kohë në cdo shtresë aplikon shkallëzueshmëri vetëm në shtresat ku koha e përgjigjes e kalon vlerat e referencës SLA/SLO.

Algoritmi i prezantuar funksionon në mënyre reaktive. Në këtë mënyrë funksionojnë shumica e algoritmeve që përdoren në ofruesit publik të Cloud. Algoritmi i propozuar në ndryshim nga veprimi automatik heq/shton instancën me parametra më të vegjël dhe riaplikon algoritmin për matjen e performancës. Në rast të mos arritjes së kohës së përgjigjes së dëshiruar atëherë algoritmi heq instancën e fundit dhe aplikon një instancë me parametra më të lartë.

Në këtë punim kemi aplikuar radhën e tipit M/M/1 për të analizuar dhe modeluar sjelljen e aplikimeve në Cloud dhe per të vlerësuar kohën e përgjigjes. Në këtë model kërkesat që paraqiten përcaktohen nga procesi Poisson.

Algoritmi i prezantuar funskionon në bazën e VM duke ulur ose ngritur numrin e makinave virtuale. Ai vepron si në aplikimet shumë shtresore por edhe në sistemet e veçuara. Në eksperimentin tonë ne jemi bazuar edhe në menaxhimin e makinave virtuale nëpërmjet mjeteve të shkallëzueshmërisë të gjendura në Eucalyptus dhe më saktë në ñeuca2oolsò. Këto na e bëjnë të mundur aplikimin e sa më të detajuar dhe monitorimin sa më të mirë të shkallëzueshmërisë.

Për të aplikuar ñScale-Upò ofrohen 3 tipe të ndryshme metodash shkallëzueshmërie me nivele të ndryshme prioriteti. Metodat vetë riparuese dhe me shkallëzueshmëri burimesh kanë prioritetet në raport me të tjerat. Të dyja këto shtojnë burimet e makinës virtuale duke përdorur burimet e makinës fizike. Koncepti i vetë riparimit ndodh në rastin kur dy makina virtuale të të njëjtit tip apo konfigurimi ndodhen në të njëjtin server fizik dhe kur burimet fizike në këtë makinë janë të mbingarkuara. Në rastin tjetër kur makina fizike (serveri) kanë burime të mjaftueshme atëherë makinës virutale i vihen në dispozicion burime të disponushme dhe të papërdorura duke përmirësuar dhe ruajutr parametrat e SLO/SLA.

Në të njëjtën mënyrë dhe në ñScale-Downò qëllimi është të hiqen sa më shumë makina virtuale por me qëllim për të mbajtur nivelet e SLA/SLO brenda parametrave të paracaktuar. Me heqjen e makinës virtuale koha e përgjigjes do të rritet por monitorimi i

vazhdueshëm do të jetë, kështu që kjo kohë do të mbetet brenda parametrave të SLA/SLO. Heqja e një makine virtuale do të ulë dhe kostot operacionale.

5.6.1 Modelimi i serverave në bazë të kërkesave dhe kohës së përgjiegjes

Për sa i përket modelimit të serverëve do të duhet që të bëjmë një profilizim të serverëve sipas kërkesave për të parë nivelin maksimal të kërkesave që mund të përballojë një server. Për bazë në këtë punim kemi marrë parasysh modelin e Markov M/M/1 sipas parimin FIFO (First In ï First Out). Kërkesat në hyrje përcaktohet si dhe raporti i shërbimit . Të dy këto parametra kanë të njëjtën madhësi (kërkesa/sec). Në rast se < atëherë sistemi nuk është stabël dhe do të duhet të kemi implementojmë ñScale-UPò.

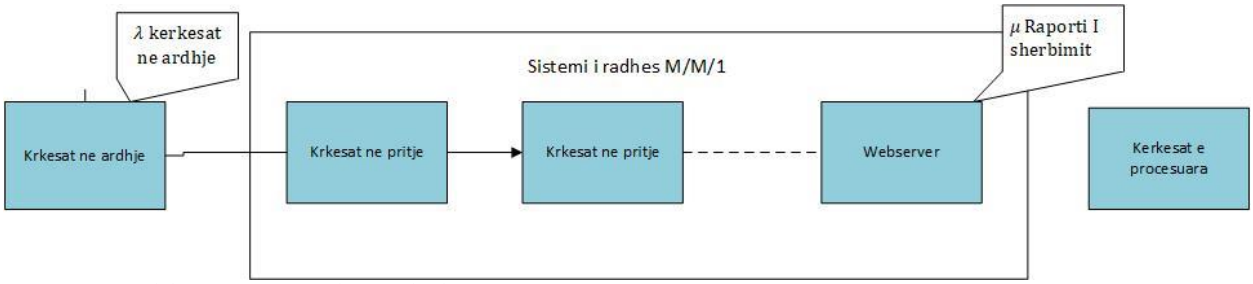


Figura 5 5 Modelimi i Serverit në bazë të kërkesave

Në procesimin e kërkesave do të merret parasysh modelimi ku kërkesat do të aplikohen në bazë të parimit ñkush vjen i pari shërbehet i pariò. Kërkesat do të jenë në një radhe si në figurë. Në sistem gjendjet e kërkesave janë ose në radhe ose duke marrë përgjigje.
Numri total i gjithë kërkesave do të ishte:

$$k = \lambda \times r = \frac{\lambda}{\mu}$$

Ndërsa për të gjithë serverat që do të shërbejnë të dhënat numri i kërkesave do te jetë sa shumatorja e kërkesave në sistem Për dy ose më shumë servera situata do të ishte:

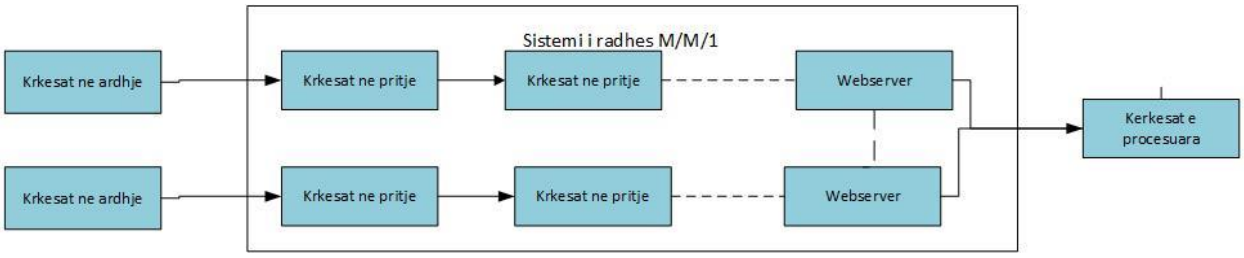


Figura 5 6 Modelimi i dy Serverave në bazë të kërkesave

Në modelimin e shtresave në aplikim do të marrim të mirëqenë që serverat e të njëjtës shtrese kanë të njëjtë raport shërbimi si dhe që kërkesat shpërndahen në mënyrë të barabartë në të gjithë serverat e shtresës. Të gjitha kërkesat kalojnë nëpërmjet LoadBalancers që menaxhojnë shpërndarjen e kërkesave.

Numri i kërkesave në shtresë do të ishte *n x ?.* ku n është numri i serverëve. Nga algoritmi koha e përgjigjes llogaritet vetëm atëherë kur kemi ndryshime në shtresën e serverëve, pra shtim ose heqje të serverëve, dmth ekzekutim të ñScale-UPò ose ñScale-DOWNò.

Koha e përgjigjes r(s) për një server do të jetë koha e nxjerrjes dhe paraqitjes së informacionit. Në algoritëm do të dallojmë dhe në cilën shtresë kemi kohë vonesë dhe ku duhet të ndërhyjmë për shkallëzueshmëri. Të gjitha kërkesat e përdorueseve kalojnë nëpërmjet LoadBalancers. Koha e përgjigjes totale e aplikimit është sa shumatorja e kohëve të përgjigjes në çdo shtresë.

Para se algoritmi të ndërmarrë ndonjë shkallëzueshmëri sigurohet që nuk ka procese shkallëzueshmërie të pa përfunduara.

Nëse koha e përgjigjes e matur është më e vogël se ?? e paracaktuar për një interval të caktuar atëherë algoritmi bën një “Scale-Down” në shtresë sipas një modeli paraashikues. Ky model ilustrohet si më mirë sipas një modeli regresiv ku parashikohet në një moment të caktuar numri i serverëve Web dhe serverëve të bazës së të dhënave për një numër të caktuar kërkesash.

$$S_t = a_0 + a_1 k_t + a_2 k_t^2 + \epsilon_t$$

Ku k është numri i kërkesave dhe këto maten për një periudhe kohore 60 sek. Koeficientët e regresit a0, a1 etj rillogariten pasi merren të dhëna të mjaftueshme. Në bazë të tij kemi ndërtuar dhe algoritmin për implementimin e shkallëzueshmërise.

Menaxhimi i elasticitetit në një mënyrë efektive shfaqet që në vendosjen fillestare të makinave virtuale për aplikimin ashtu dhe gjatë rikonfigurimeve në jetëgjatësinë e aplikimit. Për sa i përket kostove algoritmi i propozuar zgjedh elasticitet të orientuar drejt kostove duke përzgjedhur veprimin më efektiv kur shton ose heq kapacitete dhe kur zgjedh konfigurimin e serverit.

Në rastin e një aplikimi me d shtresa dhe St do të ishte numri i kërkesave për shtresë i vlerësuar në fillim atëherë do të kemi problemin e parashikimit të sa serverëve do të duhen të komisionohen që në fillim në çdo shtresë. Në fillim gjithomonë parashikohen pak kërkesa kështu që cdo konfigurim do ti kënaqte kërkesat fillestare, por qëllimi është zgjedhja e konfigurimit me qeranë më të ulët.

Në algoritëm është zgjedhur një rrugë e mesme mes reagimit reaktiv ose proaktiv, për arësye të pasaktësive në parashikimin e kërkesave të mundshme.

Pavarësisht parashikimit që bën algoritmi ne e dimë paraprakisht se sa do të jetë numri i kërkesave në një moment të caktuar. E rëndësishme është profilizimi i serverëve (makinave virtuale) me qëllim kuptimin se sa kërkesave mund tôiu përgjigjet një server me një konfirgurim të caktuar.

Për testimin e kapaciteteve të serverëve kemi përdour Quelea [73] dhe Mysql Sysbench [74]. Sipas kohës së përgjigjes se kur duhet të zbatojmë shkallzueshmëridо të mundësojmë klasifikimin e serverëve.

Në përgjithësi rimodelimi i makinës virtuale duke i ndryshuar ndonjë nga parametrat në konfigurimin e makinës virtuale do të ishte opsion i mirë dhe që mund të implementohej shpejt, por duke supozuar që ne duam një shfrytëzim maksimal të makinës virtuale atëherë do të kemi që CPU ñIdle timeò do të jete shume e vogë ose përdorimi i memories do të jetë në nivelet e larta pranë saturimit dhe ndryshimi i parametrave në vetvete nuk do të ishte opsion. Atëherë opsion do të ishte krijimi i një makine virtuale shtesë.

Shkallëzueshmëria në sistem ose më mirë të themi reagimi në sistem ka të bëjë shumë se sa shpejt merren dhe interpretohen të dhënat monitorues nga pjesët e tjera të ndërvaruara.

KAPITULLI VI

Konceptimi i ambientit të testimit dhe implementimi

6.1 Hyrje

Në këtë kapitull do të japim detajet dhe implementimin e ambientit për të mundësuar testimin e algoritmit të propozuar në paragrafin e mëparëshëm si dhe prezantimin e rezultateve të marra nga implementimi. Këto rezultate do të krahasohen me rezultatet e marra në implementimin e platformës referncë Opennebula.

6.2 Zgjedhja e aplikimit

Si aplikim për të simuluar algoritmin kemi zgjedhur zgjedhur Rice University Bidding System RUBiS [75]; RUBiS është një prototip i ngjashëm me faqen ebay.com (faqe blerjesh, explorimit dhe ofertimi produktesh në Internet). Shpeshherë ky produkt përdoret për vlerësimin e projektimit të aplikimeve si dhe performancëen e serverëve të aplikimeve.

Ky produkt ofron funksionalitetet kryesore të një faqe ankandi: shitje, shfletimin dhe ofertimin. Ka tre lloj sesionesh që mund të kryhen në aplikim vizitor, blerës apo shitës. Vizitorët nuk kanë nevojë për regjistrim në aplikim ndërsa blerësit dhe shitësit po.

RUBiS përdor MySQL/MariaDB si server për bazat e të dhënave. Në vendosjen e aplikimit do të implementojmë dhe Amoeba [76] dhe Apache Ant[77]. Amoeba do të na shërbejë dhe për të të ndarë shkrimet nga leximet në bazën e të dhënave.

Amoeba, është një program me burim të hapur. Ajo funksionin për Mysql/MariaDB si një shtresë Proxy kur baza e të dhënave është e shpërndarë. Ai

vepron si funksion rrugëzimi SQL kur aksesohet MySQL në shtresën e aplikacionit dhe përqendrohet në zhvillimin e shtresës së shpërndarë të Proxy Database. Ndodhet ndërmjet klientit dhe serverit DB dhe është transparent për klientin. Siguron balancimin e ngarkesës, disponueshmëri të lartë, filtrim SQL, ndarje leximi / shkrimi, rrugëzim drejt objektivit të përcaktuar Me Amoeba mundësohet rritja e disponueshmërisë së të dhënave si dhe shpërndarja e ngarkesës në shtresën e bazave të dhënave në aplikimin tonë.

6.3 Platforma Cloud e testimit

Për implementimin e aplikimit dhe kryerjen e testeve kemi zgjedhur platformën Eucalyptus Cloud.

Algoritmi i propozuar është implementuar në një ambient laboratorik ku për bazë janë marrë tre servera HP DL 380 G7 secili me parametra 64 GB RAM dhe 1TB Hdd dhe me nga 4 porta Ethernet Gigabit. Për lidhjen midis njëri-tjetrit është përdorur një switch Cisco 3850. SWitch është përdoru për krijimin e kanaleve të etherentit të shfrytëzuar nga serverat. Portat e rrejtit janë bashkuar logjikisht me njëra tjetrën për të dhënë undësinë e transferimit të të dhënave midis sistemeve. Shpejtësia e transferimit midis servera ve me nëri tjetrin është 4 Gbps. Në to është instaluar Eucalyptus Cloud v. 4.4.3.

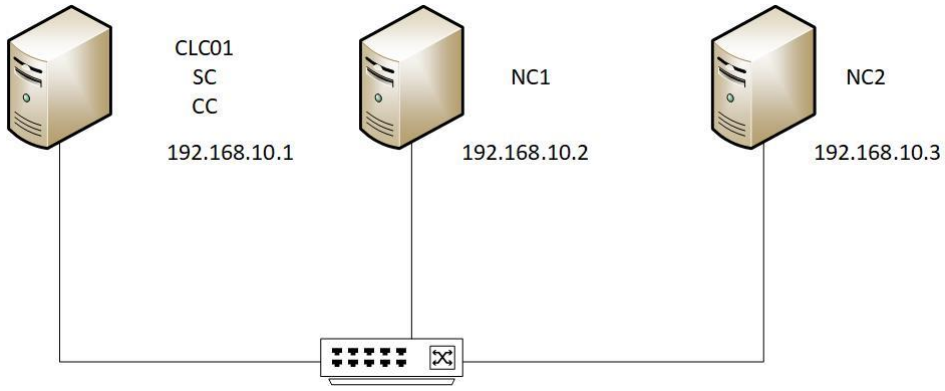


Figura 6 1 Ambienti Laboratork per implementimin e Algoritmit te propozuar

Një nga serverat është zgjedhur me rolin menaxhes të Cloud nga ku dhe ne do të menaxhojmë Cloud e implementuar, kurse dy serverat e tjerë janë vendosur si Node Controller. Në këto dy servera do të vendosim makinat virtuale.

Figura 6 2 Skema e testimit

6.3.1 Dy fjalë për Eucalyptus Cloud

Në pjesën që vijon do të paraqesim Eucalyptus, një sistem i bazuar në burimeve të hapura që përdor llogaritjen dhe infrastrukturën e ruajtjes për studime eksperimentale. Siç kemi përmendur, arkitektura e duhur virtuale vazhdon të përbëjë një fushë studimi [78] dhe analizimi, optimizimi dhe kuptimi i performancës së sistemeve të virtualizuara [79], [80] dhe [81] është një fushë aktive e kërkimit.

Përveç Cloudeve komerciale si Amazon EC2/S3, Google AppEngine etj që kemi përmendur në këtë studim që kanë një infrastrukturë më ndërfaqe të hapur ka edhe shume projekte dhe studime të cilat merren me studimin e vënies në dispozicion të burimeve nëpërmjet virtualizimit.

Usher [87]	Kornizë e menaxhimit të hapur të makinave virtuale për studiuesit
Workspaces [83]	Sistemi i bazuar në Globus [84] për të siguruar hapësira pune (d.m.th Makinat Virtuale) e cila përdor disa zgjidhje ekzistuese të zhvilluara në grid computing
Projekti i Cluster-on-demand [85]	Ofrimi i makinave virtuale për aplikimet shkencore kompjuterike

oVirt [86]	Një paketë të plotë më mjetë të integruara për menaxhimin e makinave virtuale të bazuar në Web
Open Nebula [91]	I ofron përdoruesve me disa skenarë të ndryshme konfigurimit duke lejuar një mënyrë të thjeshtë dhe fleksibël për të hartuar dhe menaxhuar drejtimin e Makinave Virtuale

Tabela 6 1 Platformat me Burim të Hapur për Menaxhimin e Cloud

Secili prej këtyre projekteve ofron një platformë për menaxhimin të hapur të burimeve. Karakteristika kryesore e Eucayptus është se imiton funksionet e Amazon Web Services duke ripërsëritur si EC2 dhe S3. Për këtë arsye, Eucalyptus mund të përdoret për të ofruar të njëjtin lloj shërbimesh për dhënien me qira të burimeve dhe të hapësirës në disk. Pra, Eucalyptus është krijuar për tu instaluar sa më lehtë duke mos pasur nevojë për burime të caktuara për tu vendosur. Gjithashtu është bazuar në një sistem të lartë modular dhe ndërfaqja e jashtme bazohet në API që është e zhvilluar nga Amazon. Një gjë e veçantë e Eucalyptus-it në krahasim me software e tjerë është ofrimi i një shtrese rrjeti virtuale që izolon trafikun e rrjetit të përdoruesve të ndryshim si dhe lejon shfaqjen e dy ose më shumë grupe që i përkasin të njëjtës Zonës Lokale të Rrjetit (LAN).

Eucalyptus Cloud

Siç përmendëm, Eucalyptus Cloud është një sistem i hapur për implementimin e ñCloudò të tipit IaaS private ose hibrid të përshtatshme me variantet Amazon [82] , [13]. Eucalyptus Cloud zbaton IaaS që përfshin shërbimet e ofruara në shtresën e infrastrukturës (servera, routers, storage systems dhe burime të tjera) . Ai është gjithashtu përgjegjës për ofrimin e infrastrukturës së nevojshme për SaaS dhe PaaS. Platforma Eucalyptus u zhvillua ne 2008 dhe versioni i pare u bë publik në 2009. Eucalyptus ka dy versione: Enterprise Edition dhe Eucalyptus Open-Source.

Arkitektura

Eucalyptus është një infrastrukturë virtualizimi qe mundëson krijimin e një Private Cloud. Komponentët qe përbëjnë Cloud ekzistojnë si një makinë virtuale. Administrimi i sistemit kryhet së brendshmi, ndërsa komunikimi me jashtë behët (nëpërmjet HTTP ose API)

përmes një nyje përgjegjëse që vepron si firewall. Figura më poshtë ilustron arkitekturën dhe ndërfaqen e komunikimit të Eucalyptus.

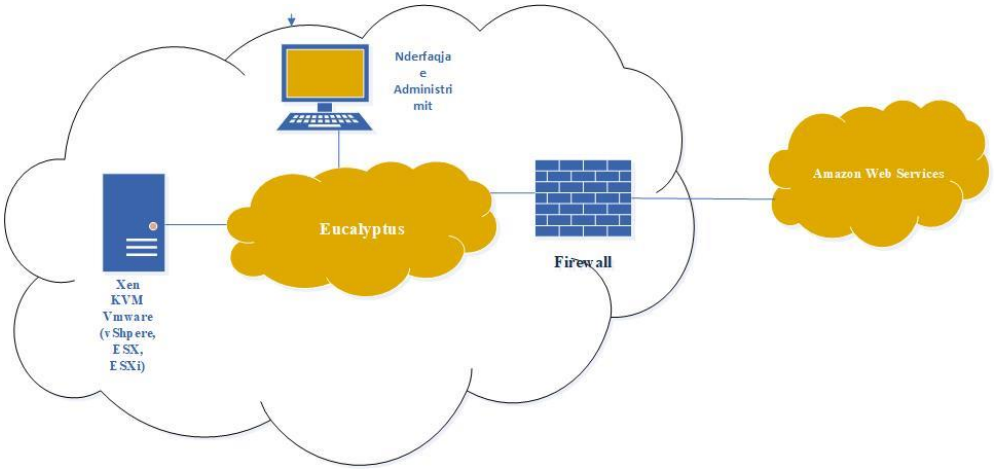


Figura 6 3 Arkitektura e Eucalyptus

Përdoruesit e Eucalyptus kanë akses në burime të ndryshme si në grupe , në kompjuter ose server. Për shkak se adresat IP janë të pakta dhe aksesi i publikut në të gjithë aplikimet të vendosura në Cloud mund të jetë e rrezikshme, zgjidhjet një nyje që mundëson aksesimin e këtyre aplikimeve. Pra aplikimet e brendshme komunikojnë më njëri tjetrin nëpërmjet një rrjeti privat dhe me nyjen që është i aksesushëm nga jashtë. Komponentët që përbëjnë arkitekturën e Eucalyptus janë: Node Controller (NC), Cluster Controller (CC), Cloud Controller (CLC) dhe Walrus Storage Controller. Në figurën e mëposhtme ilustrohet kjo arkitekture dhe këto komponentë.

Node Controller (NC)

Node Controller është moduli ku janë të vendosur makinat virtuale dhe menaxhon rrjetin virtual. Pra, Node Controller është komponenti që mundëson burimet fizike të një makine virtuale. Kështu, ne mund të caktojmë në një Node Controller një makinë fizike të vetme dhe gjithashtu mund të caktojmë disa Node Controller në makinat virtuale. Node Controller është përgjegjës për menaxhimin e gjithë ciklit të jetëgjatësisë së një makine virtuale ku përfshihet që nga nisja, monitorimi dhe përfundimi i saj. Në secilin Node Controller ka një ndërfaqe WSDL që përcakton metodat e ndryshme që ekzekutohen në të si p.sh. krijimi instance, përshkrimi instance, etj të cilat kanë përgjegjësinë për fillimin e një instance dhe

për marrjen e një raporti për burimet që po përdor instance në fjalë.

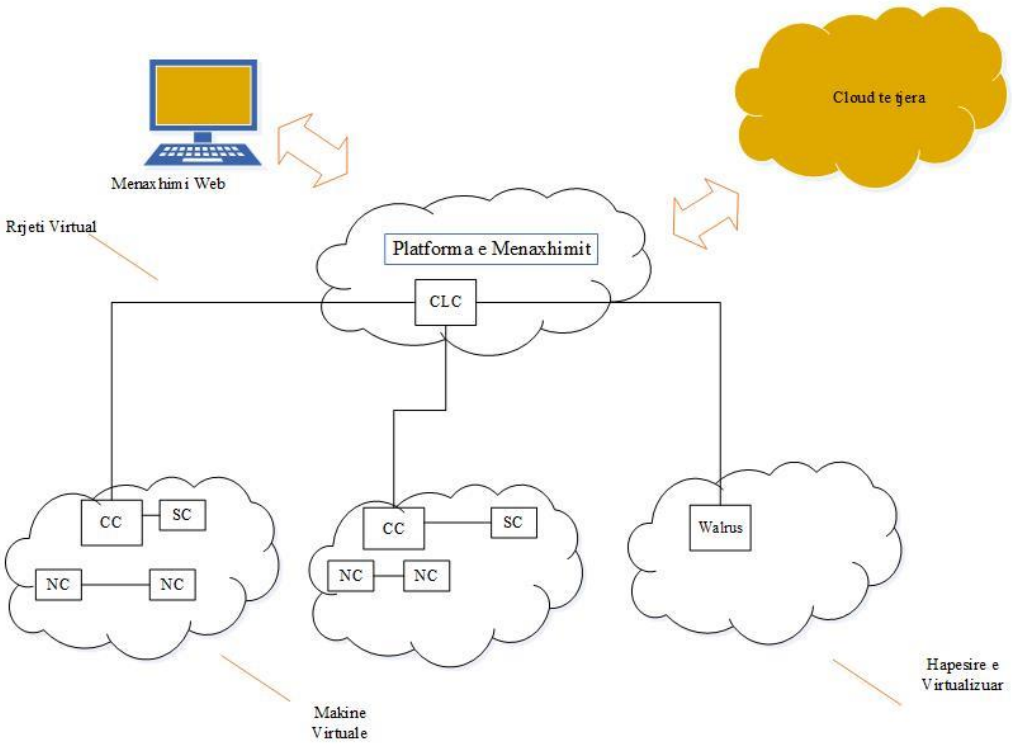


Figura 6 4 Hierarkia e Arkitekturës së burimeve të përdorura nga Eucalyptus

Cluster Controller (CC)

Cluster Controller është komponenti përgjegjës për rrugëtimin midis rrjetit vitual (Public) dhe atij të brendshëm (Private). Çdo modifikim që kryet tek burimet fizike të alokuara në Node Controller duhet të menaxhohet nga Cluster Controller. Për këtë arsye, Cluster Controller ruan një ndërfaqe WSDL me të gjitha përcaktimet e ndërfaqes që të kryejë veprimet e nevojshme. Ndërfaqja WSDL është shume e ngjashme me Node Controller por kjo ju mundëson të ndërmerrni veprime si në një Node Controller të vetëm ashtu edhe në disa prej tyre. Kur Cluster Controller merr një kërkesë për ekzekutimin e instance, ai vlerëson si një Node Controller performon këtë instance të re duke u bazuar në disa kritere të tilla si: burimet e disponueshme dhe energjia e harxhuar nga makina fizike e lidhur me Node Controller.

Për Cluster Controller është përdorur gjuha e programimit C dhe është sistemi i menaxhimit të Eucalyptus Cloud. Ai komunikon me Node Controller dhe Storage Controller. Ai menaxhon instancat dhe SLA për Custer-at.

Cloud Controller (CLC)

Cloud Controller (CLC) është porta hyrëse në Cloud për administratorët, zhvilluesit dhe përdoruesit. Pra, Cloud Controller është përgjegjës për të monitoruar Control Node për të marrë informacion në lidhje me burimet fizike, me monitorimin e performancës së programit dhe implementimin e tij nëpërmjet kërkesave të kontrollit të Cluster. Cloud Controller është i përbërë nga një sërë shërbimesh që menaxhojnë kërkesat e përdoruesve, verifikimi, qëndrueshmëria e sistemit, dhe të dhënat e përdoruesit. Këto shërbime konfigurohen dhe menaxhohen nga Enterprise Service Bus (ESB). Komponentët, që përbëjnë makinat virtuale, Service Level Agreement (SLA) dhe interface, kanë të përcaktuara mirë komunikimin e tyre brendshëm dhe i gjithë ky organizim kryhet nga ESB. Pra, zbatimi i Cloud Controller mundëson punimin në Amazon EC2 duke përdorur shërbimet web dhe ndërfaqet kërkuese.

Storage Controller

Storage Controller është komponenti i ngjashëm me Amazon Elastic Block Store i cili është një gjendje të ndërveprojë me sistemet e ruajtes, të tilla si NFS, iSCSI etij..Ky modul komunikon me Cluster Controller dhe Node Controller dhe menaxhon volumet dhe imazhin për makinat virtuale ne cluster në të cilin është i aktivizuar. Nëse një instance ka nevojë për të shkruar të dhëna jashtë Cluster atëherë mund të përdoret Walrus
i cili është i disponueshëm në çdo instance të Cluster.

Walrus

Walrus është komponenti që ofron të njëjtin shërbim si Amazon S3. Walrus i lejon përdoruesve: (1) të përdorin Walrus si një transmetues të dhënash si brenda ashtu edhe jashtë Cloud ashtu edhe për instancat që kanë filluar në nyje (2) të ruajnë të dhëna dhe të

organizojnë ato në skedarë. Pra Walrus u ofron hapësirë makinave virtuale duke ofruar një ndërfaqe HTTP për të marrë ose për të vendosur skedarë. Në të mund të vendoset çfarëdolloj skedari këtu përfshirë dhe imazhe makinash virtuale.

Monitorimi në Eucalyptus

Monitorimi në Eucalyptus realizohet në mënyra të ndryshme dhe në nivele të ndryshme të infrastrukturës. Në NC dhe CC kemi disa komanda si p.sh përshkrimi i situatës, përshkrimi i burimit, përshkrimi i rrethanave që lejon mbledhjen e dhënave në lidhje me instancat dhe burimet në një nyje ose në një grup [88]. Në kontrollin standard të sistemit operativ si CPU, sistemi, memoria, disku i memories, adresat IP bëhen nga agjenti monitorues [90]. Forma raportuese e monitorimit mund të gjenerohet në nivelin e CLC për informacione në lidhje me instancat, volumet, kapacitetet, ñsnap-shotò dhe elasticiteti i IP. Eucalyptus ka sistemin e tij të monitorimit të shërbimeve, CloudWatch [91], që siguron monitorimin e instancat për volumet e ruajtjes së bllokut elastik, monitorimin e LoadBalanacer. Pikërisht, sistemet e alarmit të CloudWatch ndihmojnë në marrjen e vendimeve në lidhje me shkallëzimin i burimeve në bazë të kërkesave [91].

6.4 Rezultatet

Në implementimin e testeve do të na ndihmojnë dhe mjete të ndryshme si ñhttperfò[7.14]. Do të gjenerojmë në numër të caktuar kërkesash për një interval të caktuar kohor në mënyre që të shohim funksionimin e algoritmit.

Për sa i përket kostove referencë sic thamë që në fillim do ti marrim empirike, ndërsa arkitektuarat midis makinave janë si më poshtë:

Tipi Instances	Virtual CPU	Disk	Memorie	Kosto*
m1.small	1	5	256	0.085 USD/Ore
m1.medium	1	10	512	0.11 USD/Ore

m1.large	2	10	512	0.17 USD/Ore
m1.xlarge	2	10	1024	0.22 USD/Ore
m3.xlarge	4	15	2048	0.315USD/Ore
m3.2xlarge	4	30	4096	0.632USD/Ore

Tabela 6 2 Referenca e Kostove

* Kostot janë emprike. Për instancat m3.xlarge dhe m3.2xlarge çmimet janë marrë nga faqja e Amazon.

Në instancat e përdorura do të aktivizojmë monitorimin e burimeve dhe konkretisht do të monitorojmë parametrat e CPU dhe memories RAM. Janë pikërisht CPU dhe memoria RAM ato që do të mbingarkohen më shumë me rritjen e kërkesave. Ky është efekti që përgjithësisht ka çdo aplikim në një server. Hapësira në disk është një dimension tjetër që do të shihej në rast se do të kishim një implementim afatgjatë. Eucalyptus ofron në vetvete monitorim të instancave dhe LB që vendosen në të. Vetë LB nëse do të aktivizojmë monitorimin në to, na japin një informacion të saktë mbi ñlatencyò. Ky parametër është i rëndësishëm për kohën e përgjigjes për të matur SLA. Është SLA që përbën dhe thelbin e algoritmit tonë,

Për kërkesat e paraqitura koha e përgjigjes do të jetë një vlerësim i përafërt në rastin e shtimit ose heqjes se një serveri në një shtresë dhe kur gjendja të jetë stabël do të merret paraysh koha e matur e përgjigjes në kohë reale. Të gjitha kërkesat e paraqitura në aplikim do te jenë N_K për një server ku n është numri i serverave ku kemi implementuar aplikimin. Të gjitha kërkesat paraqiten në LoadBalancer.

Si element për monitorimin e sistemit dhe të Loadbalancers janë përdorur Cloudwatch qe ofrohet ne Eucalyptus Cloud vetë.

Në këtë pjesë do të japim rezultatet e eksperimentit në lidhje me aplikimin e shkallëzueshmërisë në aplikim për të pare efektet ñScale-Upò dhe ñScale-Downò. Do të gjenerojmë kërkesa aplikimit dhe do ti zgjasim në kohë duke rritur numrin në mënyrë

konstante. Kontrolli dhe njëkohësisht dhe ndryshimi i kërkesave në aplikim bëhet cdo 120 sek.

Gjatë kësaj kohe do të monitorojmë numrin e makinave virtuale të krijuara si dhe performancën toatale në të gjitha shtresat në bazë të kerkesave të gjeneruara nëpërmjet httperf.

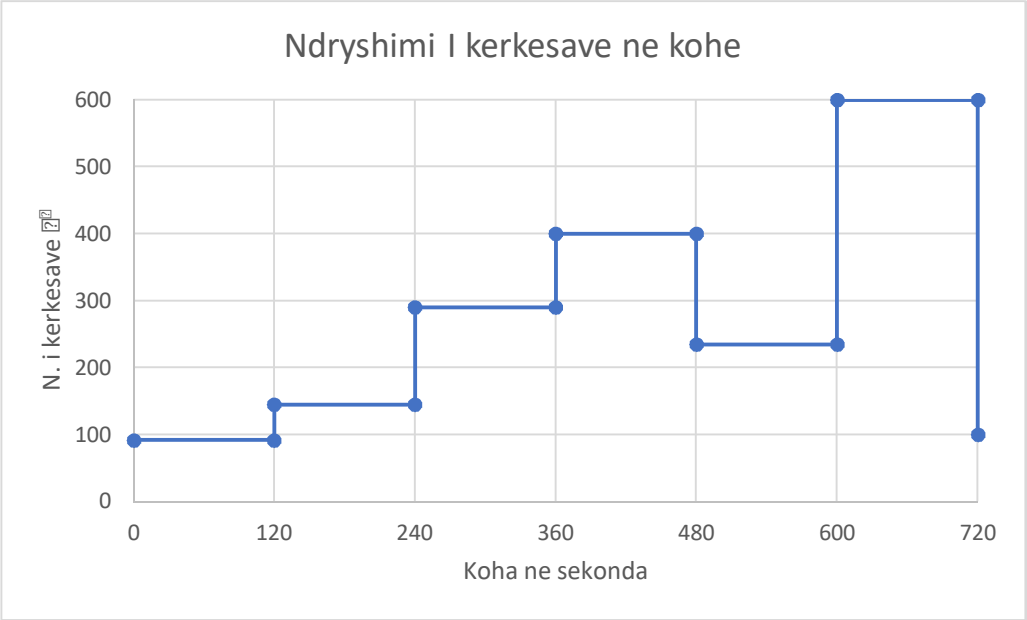


Figura 6 5 Numri i kërkesave

Koha e monitorimit do të merret e njëjta 12. Fillimisht në sistem do të vendosen nga 1 makinë vituale për shtresë dhe 2 LB (Klasik LB dhe Amoeba). Në një makinë virtuale kemi vendosur Apache dhe Tomcat ndërsa në tjetrin MySQL Master. Nyjet e tjera të MySQL do të jenë ñSlaveò.

Pas egzekutimit të kërkesave nëpërmjet ñhttperfò rezultatet do ti paraqesim më poshtë në trajtë grafike dhe do ti krahasojmë. Egzekutimet e komandës po i japim si më poshtë:

```
httperf --hog --server 10.10.10.2 --uri "/" --num-conn 12000 --num-call 1  --timeout 5 --rate 100 --port 80
&&
httperf --hog --server 10.10.10.2 --uri "/" --num-conn 17400 --num-call 1  --timeout 5 --rate 145 --port 80
&&
httperf --hog --server 10.10.10.2 --uri "/" --num-conn 34800 --num-call 1  --timeout 5 --rate 290 --port 80
&&
```

httperf --hog --server 10.10.10.2 --uri "/" --num-conn 34800 --num-call 1 --timeout 5 --rate 290 --port 80 &&

httperf --hog --server 10.10.10.2 --uri "/" --num-conn 48000 --num-call 1 --timeout 5 --rate 400 --port 80 &&

httperf --hog --server 10.10.10.2 --uri "/" --num-conn 28200 --num-call 1 --timeout 5 --rate 235 --port 80 &&

httperf --hog --server 10.10.10.2--uri "/" --num-conn 72000 --num-call 1 --timeout 5 --rate 600 --port 80

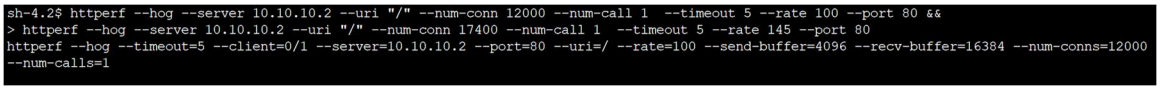


Figura 6 6 Egzekutimet e komandes

Për aksesimin e RUBiS do të duhet që të lejohet aksesi në portën TCP 80 në ñSecurity Groupò i cili është lidhur me LoadBalancer.

Më poshtë do të japim një përmbledhje, Njëkohësisht në eksperiment do ta realizojmë me kushte të barabarta, që do të thotë përdorim i njëjtë makinash virtuale me të njëjtë sasi memorie dhe procesor për të gjithë eksperimentet.

Njëkohësisht në algoritmin tonë si monitorues do të meren të gjthë parametrat e monitorimit që ofron vetë Eucalyptus Cloud.

Duke qenë që instancat m1.small, m1.medium, m1.large i kanë parametrat të ulët dhe limitet e thrershhold do ti arrinin shumë shpejt dhe algoritmi i elasticitetit do të futej shume shpejt në punë do të përdorim vetëm instancat m1.xlarge, m3.xlarge, m3.2xlarge.

Pikësëpari do të profilizojmë këto modele. Profilizimin e serverëve do ta bëjmë me Quelea [73]dhe TPC-W [89]. Kjo për të parë dhe diferencat midis mjeteve që testojnë aplikimet ueb. Pasi u kryen testet rezultatet janë si në grafikun si më poshtë:

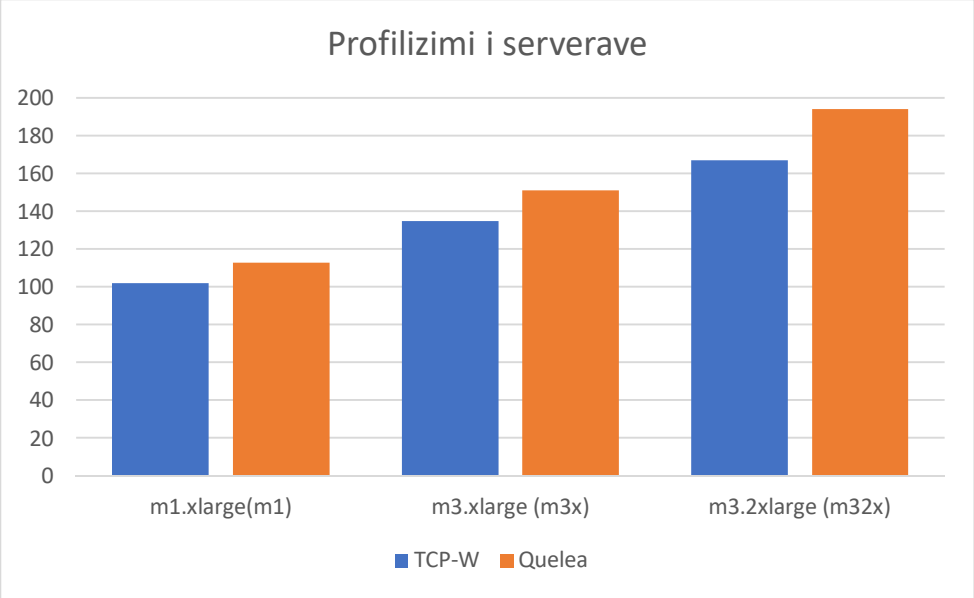


Figura 6 7 Profilizimi i Serverave me mjetet e "benchmark"

Sic shohim dhe nga grafiku kemi rezultate të ndryshem dhe kur kemi aplikuar RUBiS Gjatë testimeve u monitorua dhe niveli i përdorimit të CPU dhe u vu re që niveli maksimal i përdorimi të CPU sipas benchmark ishte përafërshisht 82% për testin Qeulea dhe 84% për testin TCP-W pra rreth 13% performancë më e mirë.

Sidoqoftë do të kemi kohën e përgjigjes që do të na shërbejë si referencë për algoritmin dhe që do të na informojë se kur do të duhet implementuar shkallëzueshmëri.

Për sa i përket kohës së përgjigjes Quelea na e jep mundësinë e matjes së mesatares së kohës së përgjigjes. Kohën e përgjigjes do ta testojmë me

Instanca	**Koha mesatare e përgjigjes nga Quelea.**
M1.large	0.31
M3.xlarge	0.24
M3.2xlarge	0.16

Tabela 6 3 Koha e përgjigjes mesatare e matur me Quelea.

Sic shohim sa më të lartë parametrat e instancës aq me e ulët është koha mesatare e përgjiegjes.

Si kohë përgjigje të dëshiruar nga aplikimi do të vendosim [illegible] = 1.2 sek. Në sistemin e monitorimit kemi konfiguruar alarme për rastet e shkeljeve të kohës së përgjigjes. Në sistem gjithashtu kemi vendosur një limit prej 10 serverash si limit i sipërm Pas aplikimit të testeve rezultatet do ti kemi si më poshtë:

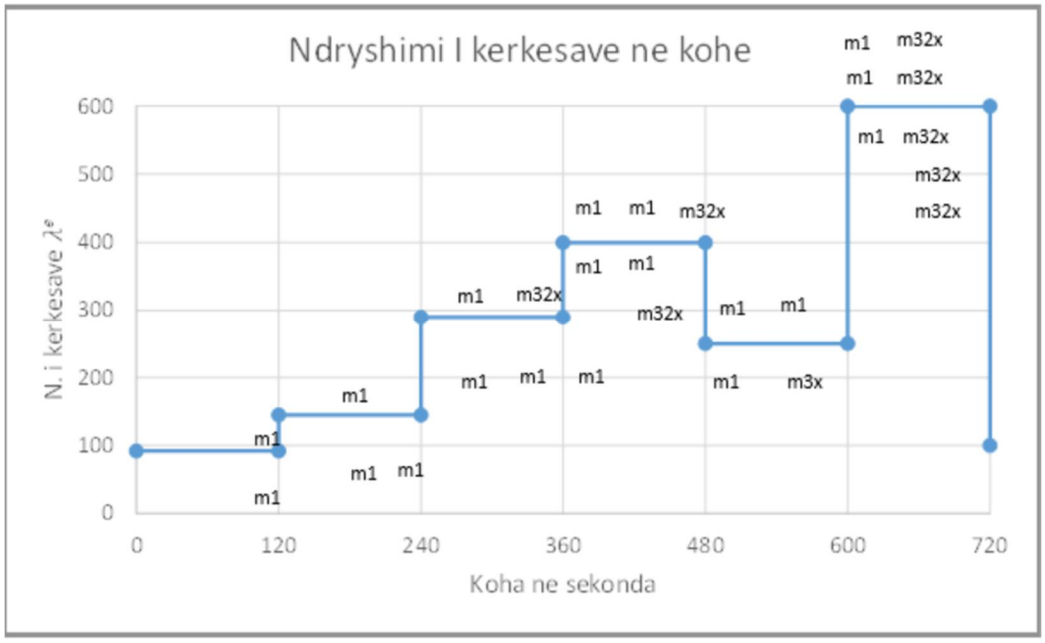
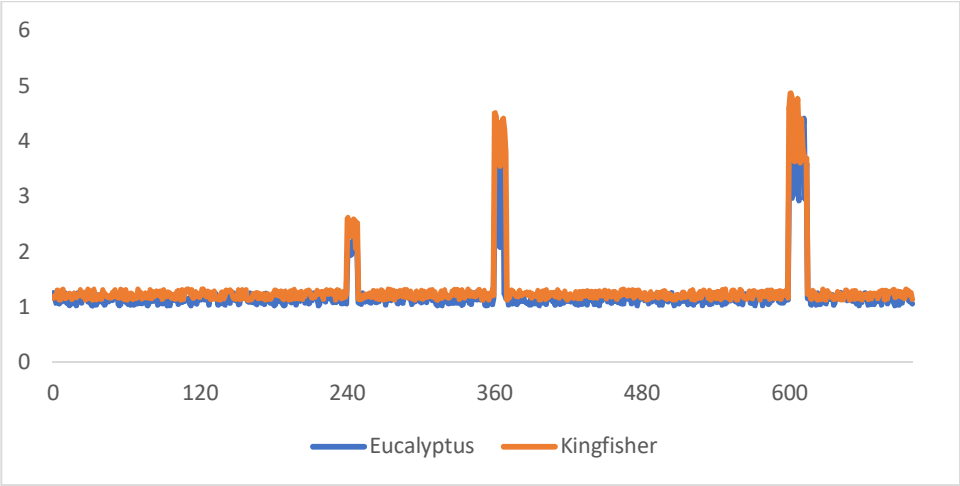


Figura 6 8 Progresimi në shtresën e Webserver

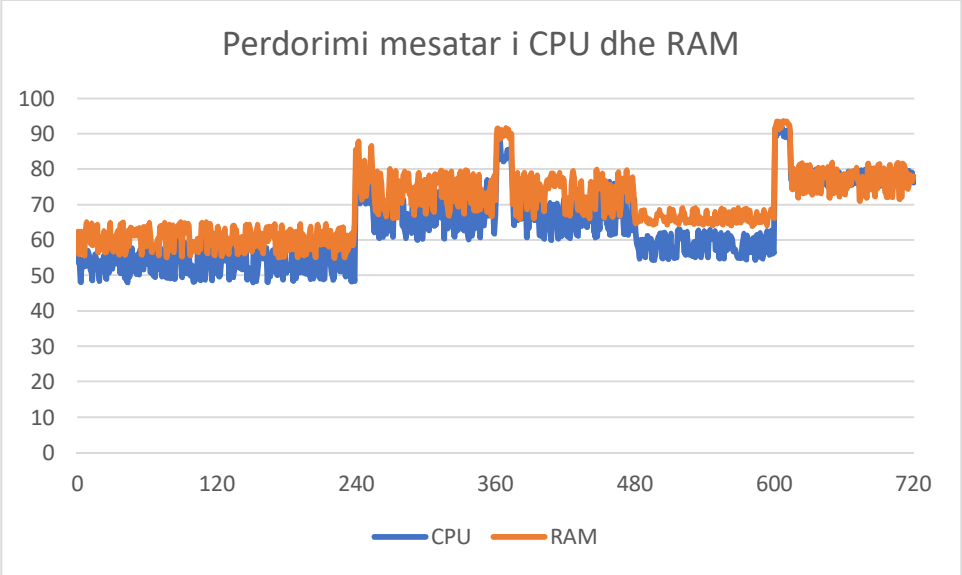
Ndërsa për stresën e bazave të dhënave grafiku i tranzicionit në këtë shtresë jepet si më poshtë:

Për sa i përket kohës së përgjigjes grafiket e kohës së përgjigjes të marra nga sistemi janë pothuajse të njëjta me ato të marra nga implementimi në Eucalyptus Cloud. Këto po i japim në grafikun më poshtë:



Figurë: Kohea e përgjigjes e marrë nga Load Balancer

Në figurën e mëposhtëme jepet përdorimi mesatar i CPU dhe RAM gjatë skenarit të aplikimit të eksperimentit. Sic e shohim dhe në figurë nga korrelimi i të dhënave përdorimi i RAM është pak më i madh se i CPU, kjo për faktin sepse edhe baza e të dhënave edhe Apache Web Server konsumojnë më shumë tabela memorie me rritjen e lidhjeve drejt aplikimit.



Figurë: Perdorimi mesatar i CPU dhe RAM

Në të dy rastet është aplikuar mekanizmi i ñLive migrationò i makinave virutale sipas skenarit ñpre-copyò[101] duke mundësuar që aplikimi të jetë i disponueshëm për gjatë gjithë testimit.

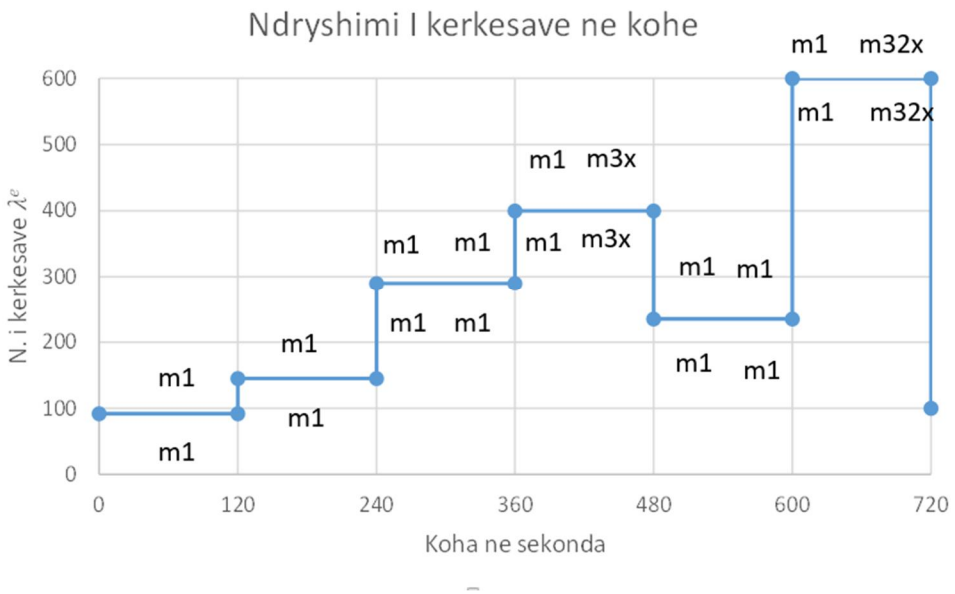


Figura 6 9 Progresimi në shtresën e bazave të të dhënave

Në të dy rastet është aplikuar mekanizmi i ñLive migrationò i makinave virutale sipas skenarit ñpre-copyò[101] duke mundësuar që aplikimi të jetë i disponueshëm për gjatë gjithë testimit.

Sic shohim nga të dy grafiqet ndryshimin më të madh e kemi tek shtresa parësore e ueb. Është aty ku paraqiten kërkesat e përdorueseve, në raastin tonë të gjenerura nga mjeti ñhttperfò. Sipas [59] shkallëzueshmëria e aplikuar në bazë të profilizimit të serverëve funksion në mënyrë të drejtë mkohën e përgjigjes së pritur. Në eksperimentin tonë kohën e përgjigjes e kemi marrë nga parametrat e konfiguruar nëpërmjet CloudWatch dhe shohim që ato përputhen me testimet e kryera me Quelea. Nga loget e sistemit vumë re një diferencë të vogël në veprimet e algoritmeve në vetvete. Si në figurën 6.7. Sipas supozimit tonë kjo vonesë nuk vjen nga shpejtësia e egzekutimit të algoritmit në vetvete por nga shpejtësia e marrjes dhe përpunimit të të dhënave monitorues. Për sa i përket përzgjedhjes se kush është opsioni më i vlefshëm të dy implementimet bëjnë një përzgjedhje që plotëson kushtet e orientimit të algoritmit drejt kostove. Gjithashtu nisja apo dhe stopimi i makinave virtuale është në të njëjtat parametra kohorë.

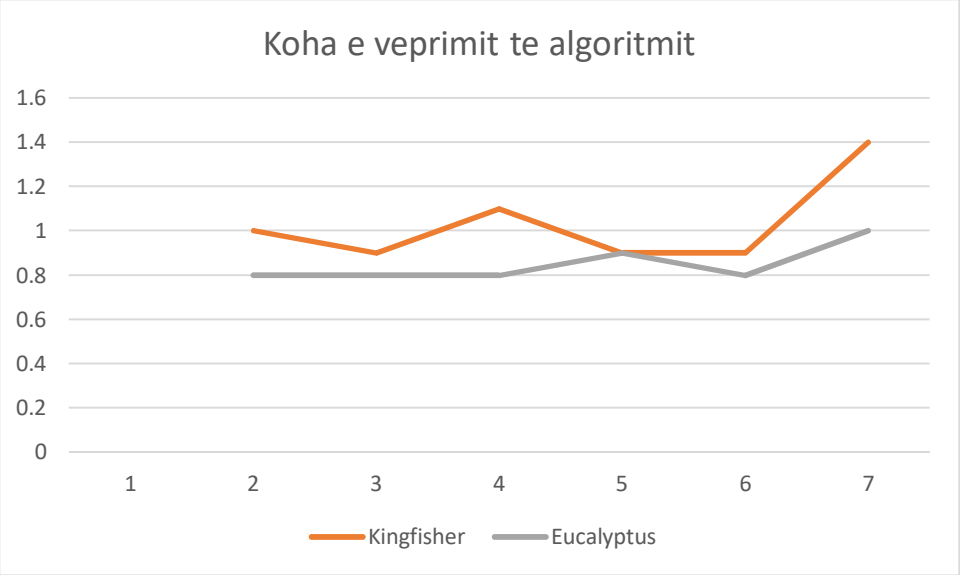


Figura 6 10 Koha e veprimti të algoritmit. Diferenca e ndryshimit të kërkesave me fillimin e reagimit të algoritmit.

Për sa i përket shtresës së bazës së të dhënave algoritmi i propozuar ashtu dhe [59] kanë të njëjtën rezultat për sa i përket shkallëzueshmërisë në shtresën e bazës së të dhënave.

Sic shohim në Figurën 6.10 koha e reagimit pra dhe e vendimmarrje në algoritmin të implementuar në Eucalyptus Cloud është më e vogël kjo dhe për shkak të marrjes së rezultateve më shpejt nga sistem i integruar i monitorimit të Eucalyptus Cloud. Ky reagim më i shpejtë është rrjedhojë e sistemeve monitoruese më të shpejta.

Për sa i përket kostove diferenca e vetme që del në shtresën e ueb sipas kostove. Sipas llogaritjeve tona gjatë gjithë eksperimentit janë përdorur instancat si më poshtë:

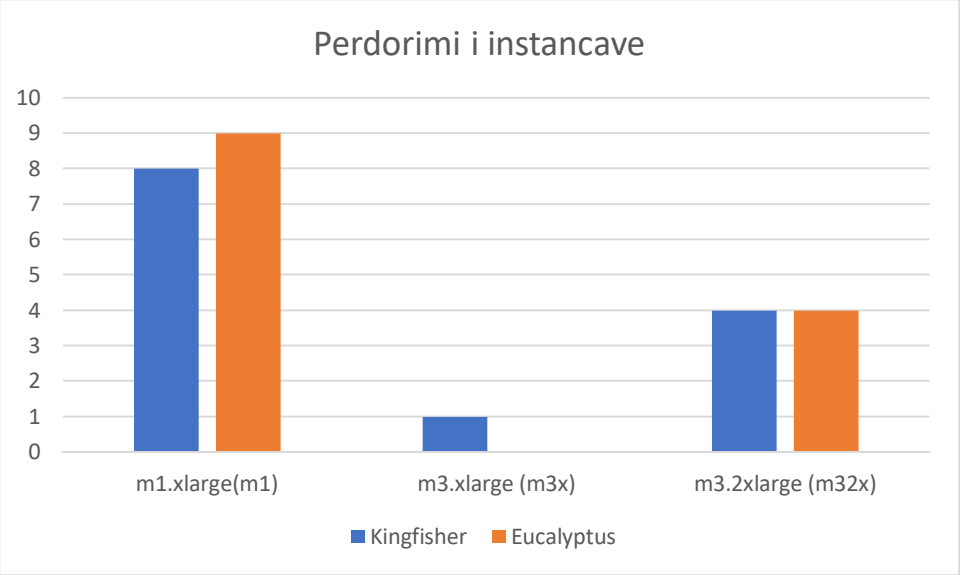


Figura 6 11 Numri i instancave të përdorura gjatë eksperimentit dhe tipet

Për sa i përket kostove:

	[59]	Eucalyptus
Kosto	0.153433 USD	0.150267 USD

Tabela 6 4 Tabela e kostove

Sic shohim kemi një kursim prej 0.003 USD në 12 min ose thënë ndryshe përafërsisht me 3% në lidhje me algoritmin e implementuar [59]. Nga kjo nxjerrim dy konkluzione që janë të rëndësishme dhe për punimin tonë por dhe për qasje të tjera ku këto aspekte janë të rëndësishme.

Përdorimi i mjeteve të integruara në sistem gjithmonë ka përparësitë e veta duke qenë se prodhuesi vetë ka njëhuri më të thella drejt sistemit por dhe duke e patur të integruar monitorimin në kod mundësia e marrjes së rezultateve në kohë reale është realisht e mundur. Marrja e rezultateve në kohë reale ndihmon dhe hallka të tjera të skemës si analiza, planifikim dhe ekzekutim. Në sistemet moderne si Eucalyptus Cloud kemi të integruara të gjitha mjetet e nevojshme për implementimin e një Cloud që vetëmenaxhohet pa nevojën e komponentëve shtesë të monitorimit.

KAPITULLI VII

Përfundime

Në përfundim të këtij punimi mund të themi shërbimet Cloud janë shërbimet më moderne dhe me kosto më të ulët dhe të përballueshme nga cdo lloj biznesi pavarësisht nga madhësia, forma apo shtrirja gjeografike. Cloud është shërbim i përgjithësuar që vë burime kompjuterike dhe programative në dispozicion, të mbështetura nga sisteme faturimi të sakta dhe transparente. Në këto kushte menaxhimi i shkallëzueshmërisë së burimeve kompjuterike është kthyer në një pikë të nxehtë dhe objekt studimi në botë. Në këtë fokus edhe në këtë punim jemi munduar të hulumtojmë sa më shumë përdorimin e shkallëzueshmërisë dhe elasticitetit me qëllim ofrimin e një performance të paracaktuar në mbështetje të kërkesave ndaj një aplikimi/ baze të dhënash. Praktikisht zbatimi i teorive të vlerësimit paraprak nëpërmjet metodave probabilitare. Në këtë punim ne kemi studiuar menaxhimin e shkallëzueshmërisë dhe elasticiteti si proces shkallëzueshmërie i automatizuar si dhe teknikat që i përshtaten këtyre modeleve, si aplikimeve ashtu dhe algoritmeve ne vetvete. Kontributet e këtij punimi janë:

Elasticiteti në aplikim dhe baza të dhënash:

- Një algoritëm u implementua si shërbim në ambientin tonë test (një kopje e shërbimit Cloud) që realizon shkallëzueshmëri automatike të burimeve Cloud duke simuluar kërkesa mbi aplikim dhe bazat e të dhënave. Implementimi i algoritmit me orientim drejt kostove rezultoi në një ulje të shpenzimeve rreth 3% për periudhën e testimit.
- U propozua një shkallëzim elastik që bazohet në varësinë e kërkesave dhe performancës dhe që bën një zgjedhje për nga instancat sipas kostove dhe në përputhje me profilizimet paraprake të serverave. Në këtë implementim shkallëzueshmërie u analizuan rëniet në performancë në shtresat e aplikimit. Sipas algoritmit të prezantuar pët implementimin e shkallëzueshmërisë ulen

kostot për burimet në përdorim duke implementuar shkallëzueshmëri vetëm në shtresat ku kemi rënie performance. Algoritmi dallon se ku krijohen vonesat në çdo shtresë. Në punim u prezantua dhe optimizimi i shkallëzueshmërisë së burimeve në vetvete (CPU / Memorie) në nivel makine virtuale.

- U shfrytëzua monitorimi i brendshëm i vetë ambientit Eucalyptus Cloud dhe u pa që monitorimi i brendshëm ofron matje më të shpejta dhe saktësi më të madhe se sa zgjidhjet e treta. Një matje më e shpejtë ka si rezultat një reagim më të shpejtë. Algoritmi ka një veprim rreth 17% më të shpejtë se algoritmi i propozuar.

Studime dhe mundësi të mëtejshme:

I gjithë punimi është i fokusuar në menaxhimin e shkallëzueshmërisë dhe elasticitetit me kosto sa më të pranueshme. Menaxhimi i elasticitetit në një shtresë më të gjerë aplikimiesh do ta bënte më përgjithësues menaxhimin e shkallëzueshmërisë së propozuar. Një analizë më e thelluar në lidhje me gjithë faktorët që ndikojnë në shkallëzueshmëri do ta bënte dhe me analitik studimin duke pare dhe modelin e implementimit si dhe platformën Cloud mbi të cilën do të kryhet testimi, limitet në buxhet, modelimi i kostove (p.sh faturimi sipas të dhënave të transmetuara). Përdorimi i të gjithë elemenëve jep dhe një pamje më të plotë. Implementimi i këtyre skenarëve ka sfidat e veta:

- Në punimin tonë kosotot/qiratë për cdo makinë virtuale janë marrë fikse. Një studim më i përafërt me realitetin do të ishte një model dinamik kostosh duke patur parasysh mënyra të ndryshme tarifimi. Shumica e ofruesëve të Cloud kanë një mënyrë tarifimi fikse por ka dhe ofrues si Amazon që ka tarifim mbi instanca rezervë që kanë një çmim më të ulët se se instancat në katalog në vetvete. Amazon i ndryshon çmimet këtyre instancave në mënyrë të rregullt.

- Një studim shtesë mund të jetë që përveç kostove të ulëta mund të shikohet dhe sa ulen kostot e energjisë. Ndikimi në kostot energjitike është me efekt pozitiv tek të dy palët si tek përdoruesi i Cloud, ofruesi i Cloud dhe përdoruesi fundor.
- Shakllzueshmëria në shumë aplikime ose më shumë klientë, pra në ambient me disa ofrues shërbimesh.
- Realizimi i benchmark edhe në ofrues Cloud të tjerë si Azure ose Rackspace mund të na japë një pamje më të qartë dhe për raportin kosto/benefite në lidhje me perdormacën. Sipas [100] rezultate të ndryshëm testimi dalin në ambiente të ndryshëm Cloud. Mos njohja e harduare që ndodhet nën serverin virtual na e vështirëson dhe përzgjedhjen në lidhje me vendin se ku do të mund të vendoset aplikimi.

REFERENCAT

[1] Luis M. Vaquero, Luis Rodero-Merino, Juan Caceres, and Maik Lindner. ñA Break in the Clouds: Towards a Cloud Definitionò. In: AMC SIGCOMM Computer Communication Review, Volume 3, Issue 1 (January 2009), pp. 50ï 55. ISSN: 014 6-4833. DOI: 10 . 1145 / 1496091 . 1496100. URL: http://doi.acm.org/10.1145/1496091.1496100.

[2] Shuai Zhang and Shufen Zhang, and X. Chen and Xiuzhen Huo. ñ Cloud Computing Research and Development Trendò. In 2010 Second International Conference on Future Networks(2010), pp.93-97, URL: https://www.semanticscholar.org/paper/Cloud-Computing-Research-and-Development-Trend-Zhang-Zhang/60eb6ca2993be1330cb845e1ac60d0188f61f08f#citing-papers (accessed April 20, 2021)

[3] Interl IT Center (August 2013), ñ Planning Guide: Virtualization and Cloud Computing: Steps in the Evolution from Virtualization to Private Cloud Infrastructure as a Serviceò, URL: Virtualization and cloud computing (intel.in) (accessed: April 10, 2021)

[4] Joe Schulz, ñKey Features Of Cloud Computing, Blog, CloudTweaks,ò 03-Sep-2012. [Online]. URL: http://www.cloudtweaks.com/2012/09/key-features-of-cloud-computing/. (accessed May 15, 2021)

[5] Olive, Ch., White paper: Cloud Computing Characteristics are key, GP Strategies, URL: https://www.gpstrategies.com/wp-content/uploads/2016/04/wpCloudCharacteristics.pdf (accessed June 21, 2021)

[6] D. M. Surgient, ñThe five defining characteristics of cloud computing,ò ZDNet.(9 April 2009). URL: http://www.zdnet.com/news/the-five-defining-characteristics-of-cloudcomputing/287001. (accessed April 10, 2021)

[7] Giuseppe Aceto, Alessio Botta, Walter de Donato, and Antonio Pescap (November 2012), ñCloud Monitoring: definitions, issues and future directionsò, Conference: Cloud Networking, 2012 IEEE, (accessed April 5 2021)

[8] Mell, P. and Grance, T. (2011), The NIST Definition of Cloud Computing, Special Publication (NIST SP), National Institute of Standards and Technology, Gaithersburg, MD, [online], https://doi.org/10.6028/NIST.SP.800-145 (Accessed June 18, 2021)

[9] VB: Cloud layers. http://venturebeat.com/2011/11/14/cloud-iaas-paas-saas/.

[10] Amazon Web Services. http://aws.amazon.com/.

[11] Nimbus. http://www.nimbusproject.org/.

[12] OpenNebula. http://www.opennebula.org/

[13] Eucalyptus. https://www.eucalyptus.com/.

[14] Radu Tudoran. High-Performance Big Data Management Across Cloud Data Centers. Computer science. ENS Rennes, 2014. English. fftel-01093767f. URL:

[15] Googe Apps Engine. https://cloud.google.com/.

[16] Microsoft Azure. http://azure.microsoft.com/en-us/.

[17] Yingyi Bu, Bill Howe, Magdalena Balazinska, and Michael D. Ernst. ñHaLoop: efficient iterative data processing on large clustersò. In: Proc. VLDB Endow. 3 (1-2 2010), pp. 285ï 296. ISSN: 2150-8097.

[18] Cisco Systems, ñCisco Cloud Computing - Data Center Strategy, Architecture, and Solutionsò [Online]. URL: http://www.cisco.com/web/strategy/docs/gov/CiscoCloudComputing_WP.pdf , 2009. (accessed: March 21, 2021)

[19] KalpanaParsi and M.Laharika- A Comparative Study of Different Deployment Models in a Cloud

https://www.ijarcsse.com/docs/papers/Volume_3/5_May2013/V3I5-0229.pdf

[20] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A.

Patterson, A. Rabkin, and M. Zaharia, ñAbove the Clouds: A Berkeley View of Cloud Computing,ò University of California, EECS Department, Berkeley, California Technical Report No. UCB/EECS-2009-28, 10 February 2009. URL: http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.html (Accessed Febraury 10, 2021)

[21] Vic (J.R.), ñ Security Cloud: Cloud Computer Security Techniques and Tacticsò , 1st Edition, Syngress, (April 2011), eBook ISBN: 9781597495936, p.36

[22] MacVittie, L. (2009). Load balancing is key to successful cloud-based (dynamic) archi-tectures. DevCentral Home . Available at:http://devcentral.f5.com/weblogs/macvittie/archive/2009/01/23/load- balancing-is-key-to-successful-cloud-based-dynamic-architectures.aspx. (accessed March 21, 2021)

[23] Tushar Desai, Jignesh Prajapati, "Analysis A Survey Of Various Load Balancing Techniques And Challenges In Cloud Computing ." INTERNATIONAL JOURNAL OF SCIENTIFIC & TECHNOLOGY RESEARCH VOLUME 2, ISSUE 11, NOVEMBER 2013 ISSN 2277-8616158 IJSTR©2013

[24] Chuanpeng Li, Chen Ding, and Kai Shen. Quantifying the cost of context switch. In Proceedings of the 2007 workshop on Experimental computer science, ExpCS ð07, New York, NY, USA, 2007. ACM.

[25] N. Jain and S. Choudhary, "**Overview of virtualization in cloud computing**," *2016 Symposium on Colossal Data Analysis and Networking (CDAN)*, 2016, pp. 1-4, doi:10.1109/CDAN.**2016**.7570950.

[26] M. Singh, "Virtualization in Cloud Computing- a Study," 2018 International Conference on Advances in Computing, Communication Control and Networking (ICACCCN), 2018, pp. 64-67, doi: 10.1109/ICACCCN.2018.8748398.

[27] The NIST Definition of Cloud Computing. U.S. Department of Commerce, National Institute of Standards and Technology Special Publication 800-145 (September 2011). http://csrc. nist.gov/publications/nistpubs/800-145/SP800-145.pdf

[28] M. Varian, ñVM and the VM community: Past, present, and futureò, Office of Computing and Information Technology, Princeton, University, Princeton, NJ, 1997.

[29]Jinho Hwang, Sai Zeng, Frederick y Wu, and Timothy Wood, ñA Component-Based Performance Comparison of Four Hypervisors," 13th IFIP/IEEE International Symposium on Integrated Network Management (IM) Technical Session, 2013.

[30]Malhotra L, Agarwal D and Jaswal A, ñ Virtualization in Cloud Computingò Journal of Information technology an Software Engineeringm (2014), Vol.4 No2, URL: https://www.longdom.org/open-access/virtualization-in-cloud-computing-2165-7866-1000136.pdf (accessed April 10, 2021)

[31] Rui Han, Li Guo, Moustafa M. Ghanem, Yike Guo, Lightweight Resource Scaling for Cloud Applications, 2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing

[32] L. Badger, T. Grance, R. Patt-Corner, and J. Voas, ñDraft cloud computing synopsis and recommendations,ò NIST special publication, vol. 800, p. 146, 2011.

[33] E. F. Coutinho, F. R. de Carvalho Sousa, P. A. L. Rego, D. G. Gomes, and J. N. de Souza, ñElasticity in Cloud Computing: a Survey,ò Annals of Telecommunications, pp. 1ï 21, 2015.

[34] G. Galante and L. C. E. d. Bona, ñA Survey on Cloud Computing Elasticity,ò in Proceedings of the 2012 IEEE/ACM Fifth International Conference on Utility and Cloud Computing, UCC ô12. Ëashington, DC, USA: IEEE Computer Society, 2012, pp. 263ï 270.

[35] N. R. Herbst, S. Kounev, and R. Reussner, ñElasticity in Cloud Computing: Ëhat It Is, and Ëhat It Is Not,ò in Proceedings of the 10th International Conference on Autonomic Computing (ICAC 13). San Jose, CA: USENIX, 2013, pp. 23ï 27

[36] Nikolas Roman Herbst, Samuel Kounev, and Ralf Reussner. Elasticity in cloud computing: What it is, and what it is not. In ICAC, pages 23ï 27, 2013.

[37] DUBOC, L., ROSENBLUM, D., AND WICKS, T. A Framework for Characterization and Analysis of Software System Scalability. In Proceedings of the 6th joint meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on The Foundations of Software Engineering (ESEC-FSE ô07) (2007), ACM, pp. 375ï 384.

[38] Sotiriadis, S., Bessis, N., Amza, C., & Buyya, R. (2016). Vertical and horizontal elasticity for dynamic virtual machine reconfiguration. IEEE Transactions on Services Computing, 1ï 1. doi:10.1109/tsc.2016.2634024

[39] Yahya Al-Dhuraibi, Fawaz Paraiso, Nabil Djarallah, Philippe Merle. Elasticity in Cloud Computing: State of the Art and Research Challenges. IEEE Transactions on Services Computing, IEEE, 2018, 11 (2), pp.430-447. ff10.1109/TSC.2017.2711009ff. ffhal-01529654f

[40] G. Galante and L. E. De Bona, ñ A Survey on Coud Computing Elasticityò, Published in Proceedings of Fifth Interntional Conference on Utility and Cloud Computing (UCC) (November 2012), p.263-270
[41] Ullah, A., Li, J., Shen, Y., & Hussain, A. (2018). A control theoretical view of cloud elasticity: taxonomy, survey and challenges. Cluster Computing. doi:10.1007/s10586-018-2807-6, URL: sci-hub.do/10.1007/s10586-018-2807-6 (accessed April 10, 2021)
[42] L. M. Vaquero, L. Rodero-Merino, and R. Buyya, ñDynamically scaling applications in the cloud,ò SIGCOMM Comput. Commun. Rev., vol. 41, pp. 45ï 52, 2011.
[43] F. Paraiso, P. Merle, and L. Seinturier, ñsoCloud: A ServiceOriented Component-Based PaaS for Managing Portability, Provisioning, Elasticity, and High Availability across Multiple Clouds,ò Computing, pp. 1ï 27, 2014
[44] Amazon Web Services (2011) AWS economic center. http: // aws.amazon.com/economics/
[45] Red Hat Inc., ñScaling the LAMP Stack in a Red Hat Enterprise Virtualization Environment,ò Aug. 2009. Available: http://ëëë.redhat.com/rhecm/rest rhecm/jcr/repository/collaboration/jcr:system/jcr:versionStorage/54a4560b0a070d5442ce df2879 9bff35/1/jcr:frozenNode/rh:resourceFile
[46] B.Suleiman, S. Bakr, R. Jeffery and A. Liu, ñ On Understanding the economics and elasticity challenges of deploying business application on public cloud infrastructureò, Jisa, Vol.2, No.3, pp. 1-21, 2021.
[47] P.C. Brener, ñ Is your cloud elastic enough?: Performance modelling the elasticity of infrastructure as a service (IaaS cloud) applicationò in Proceedings of the 3rd AMC/SPEC International Conference on Performance Engineering, ser. ICPE 12, New York, USA, 2012, pp. 263-266
[48] B. Suleiman, S. Sakr, S. Venugopal and W. Sadiq , ñTrade-off analysis of elasticity approaches for cloud-based business applicationsò, in Proceedings of the 13rd International Conference on Web Information Systems Engineering, ser. Wise 12, Berlin, Heidelberg: Springer-Verlag, 2012, pp 468-482

[49] A. Naskos, A. Gounaris, and S. Sioutas, ñ Cloud Elasticity; A surveyò, Published in ALGOCLOUD 2015; URL: https://www.semanticscholar.org/paper/Cloud-Elasticity%3A-A-Survey-Naskos-Gounaris/7c8b6f35041a6499d1319f60a1d1af7e0975267c (accessed April 10, 2021)

[50] Ashraf, A., Byholm, B., Porres, I.: Cramp: Cost-effcient resource allocation for multiple web applications with proactive scaling. In: Cloud Computing Technology and Science (CloudCom), 2012 IEEE 4th International Conference on. pp. 581{586 (2012)

[51] YANG Jie, QIU Jie, LI Ying, A Profile-based Approach to Just-in-time Scalability for Cloud Applications, 2009 IEEE International Conference on Cloud Computing

[52] da Silva Dias, A., Nakamura, L.H.V., Estrella, J.C., Santana, R.H.C., Santana, M.J.: Providing iaas resources automatically through prediction and monitoring approaches. In: IEEE Symposium on Computers and Communications, ISCC 2014, Funchal, Madeira, Portugal, June 23-26, 2014. pp. 1{7 (2014)

[53] Vaquero, L.M., Mor_an, D., Gal_an, F., Alcaraz-Calero, J.M.: Towards runtime reconfiguration of application control policies in the cloud. Journal of Network and Systems Management 20(4), 489{512 (2012)

[54] Paraiso, F., Merle, P., Seinturier, L.: socloud: A service-oriented component-based paas for managing portability, provisioning, elasticity, and high availability across multiple clouds. CoRR abs/1407.1963 (2014)

[55] Claudio A. Ardagna, Ernesto Damiani, Fulvio Frati, Davide Rebeccani, Guido Montalbano, Marco Ughetti, A Competitive Scalability Approach for Cloud Architectures, 2014 IEEE International Conference on Cloud Computing

[56] Sebastian Lehrig, Hendrik Eikerling, Steffen Becker, Scalability, Elasticity, and Efficiency in Cloud Computing:a Systematic Literature Review of Definitions and Metrics, 11th International ACM SIGSOFT, 2015

[57] Sushil Kumar Sah, Shashidhar Ram Joshi, Scalability of Efficient and Dynamic Workload Distribution in Autonomic Cloud Computing, 2014 International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT)

[58] Nagamani H Shahapure, P Jayarekha, Distance and Traffic Based Virtual Machine Migration forScalability in Cloud Computing, International Conference on Computational Intelligence and Data Science (ICCIDS 2018)

[59] Upendra Sharma, Prashant Shenoy, Sambit Sahu and Anees Shaikh, A Cost-aware Elasticity Provisioning System for the Cloud, 2011 31st International Conference on Distributed Computing Systems

[60] Khalid Alhamazani, Rajiv Ranjan, Fethi Rabhi, Lizhe Wang, Karan MitraCloud Monitoring for Optimizing the QoS of Hosted Applications, CloudCom.2012

[61] Jonathan Stuart Ward, Adam Barker, Observing the clouds: a survey and taxonomy of cloud monitoring, ournal of Cloud Computing: Advances, Systems and Applications (2014)

[62] Khalid Alhamazan, Rajiv Ranjan, Karan Mitra, Fethi Rabhi, Samee Ullah Khan, Adnene Guabtni, Vasudha Bhatnagar, An Overview of the Commercial Cloud Monitoring Tools: Research Dimensions, Design Issues, and State-of-the-Art, Computing volume 97, pages357ï 377 (2015)

[63] Mahantesh N. Birje, Chetan Bulla, Cloud Monitoring System: Basics, Phases and Challenges, International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878, Volume-8 Issue-3, September 2019

[64] Rajiv Ranjan,À, Rajkumar Buyya, Philipp Leitner, Armin Haller and Stefan Tai A note on software tools and techniques for monitoring and prediction of cloud services, Softw. Pract. Exper. 2014; 44:771ï 775

[65] Giuseppe Aceto, Alessio Botta, Walter de Donato, Antonio Pescapè, Cloud Monitoring: definitions, issues and future directions, CloudNet 2012

[66] Giuseppe Aceto, Alessio Botta, Walter de Donato, Antonio Pescapè Cloud monitoring: A survey, Elsevier 2013

[67] Yun Chang Liu 1 , Chun Lin Li, A Stratified Monitoring Model for Hybrid Cloud, Applied Mechanics and Materials Vols. 719-720 (2015) pp 900-906

[68] TARIQ FALAH, ALWADA, CLOUD COMPUTING AND MULTI-AGENT SYSTEM: MONITORING AND SERVICES, Journal of Theoretical and Applied Information Technology 15 th May 2018. Vol.96. No 09

[69] Cloud Security Alliance (https://cloudsecurityalliance.org/)

[70] A. Computing, "An architectural blueprint for autonomic computing," IBM White Paper, vol. 31, 2006.

[71] T. Lorido-Botran, J. Miguel-Alonso, and J. A. Lozano, "A review of auto-scaling techniques for elastic applications in cloud environments," Journal of Grid Computing, vol. 12, pp. 559-592, 2014.

[72] C. Qu, R. N. Calheiros, and R. Buyya, "Auto-scaling Web Applications in Clouds: A Taxonomy and Survey," arXiv preprint arXiv:1609.09224, 2016.

[73] https://github.com/kayceesrk/Quelea

[74] https://dev.mysql.com/downloads/benchmarks.html

[75] GitHub - sguazt/RUBiS: Implementation of the Rice University Bidding System (RUBiS)

[76] https://sourceforge.net/projects/amoeba/

[77] https://ant.apache.org/

[78] K. Adams and O. Agesen. A comparison of software and hardware techniques for x86 virtualization. In ASPLOSXII: Proceedings of the 12th international conference on Architectural support for programming languages and operating systems, pages 2ï 13, New York, NY, USA, 2006. ACM

[79] W. Huang, J. Liu, B. Abali, and D. K. Panda. A case for high performance computing with virtual machines. In ICS ð06: Proceedings of the 20th annual international conference on Supercomputing, pages 125ï 134, New York, NY, USA, 2006. ACM

[80] M. F. Mergen, V. Uhlig, O. Krieger, and J. Xenidis. Virtualization for high-performance computing. SIGOPS Oper. Syst. Rev., 40(2):8ï 11, 2006.

[81] L. Youseff, K. Seymour, H. You, J. Dongarra, and R. Wolski. The impact of paravirtualized memory hierarchy on linear algebra computational kernels and software. In HPDC, pages 141ï 152. ACM, 2008.

[82] Burns, Christine. ñStack Wars: OpenStack v. CloudStack v. Eucalyptus.ò Network World. (June 3, 2013).networkworld.com/supp/2013/enterprise3/060313-ecs3-open-stack-269899. html?sourceWWNLE_nlt_cloud_security_2013-06-0

[83] M. McNett, D. Gupta, A. Vahdat, and G. M. Voelker.Usher: An Extensible Framework for Managing Clusters of Virtual Machines. In Proceedings of the 21st LargeInstallation System Administration Conference (LISA), November 2007.

[84] I. Foster and C. Kesselman. Globus: A metacomputing infrastructure toolkit. International Journal of Supercomputer Applications, 1997.

[85] J. Chase, D. Irwin, L. Grit, J. Moore, and S. Sprenkle. Dynamic virtual clusters in a grid site manager. High Performance Distributed Computing, 2003. Proceedings. 12th IEEE International Symposium on, pages 90ï 100, 2003.

[86] oVirt home page. http://ovirt.org/.

[87] M. McNett, D. Gupta, A. Vahdat, and G. M. Voelker. Usher: An Extensible Framework for Managing Clusters of Virtual Machines. In Proceedings of the 21st Large Installation System Administration Conference (LISA), November 2007.

[88] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L.Youseff, D. Zagorodnov, ñ Eucalyptus:An open-source cloud computing infrastructureò, Journal of Physica: Conference series 180 (2009).

[89] http://tpc.org/tpcw/default5.asp

[90] https://github.com/eucalyptus/eucalyptus/wiki/Monitoring

[91] http://www.opennebula.org/

[92] https://github.com/httperf/httperf

[93] https://en.wikipedia.org/wiki/Amazon_Web_Services#History

[94]https://www.technologyreview.com/2011/10/31/257406/who-coined-cloud-computing/

[95] https://xenproject.org/

[96] https://www.linux-kvm.org/page/Main_Page

[97] https://aws.amazon.com/ec2/faqs/

[98] https://www.mysql.com/

[99] Mahantesh N. Birje, Chetan Bulla, Commercial and Open Source Cloud Monitoring Tools: A Review, 2019

[100] Borhani, Amir Hossein ; Leitner, Philipp ; Lee, Bu-Sung ; Li, Xiaorong ; Hung, Terence-WPress: Benchmarking Infrastructure-as-a-Service cloud computing systemsfor on-line transaction processing applications, 2014

[101] Anita Choudhary, Mahesh Chandra Govil2, Girdhari Singh, Lalit K. Awasthi3, Emmanuel S. Pilli1and Divya Kapil-A critical survey of live virtual machinemigration techniques, Choudharyet al. Journal of Cloud Computing: Advances, Systemsand Applications ,2017