Digitalni identiteti u oblaku

Klobučar, Dora

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Sveučilišni studij

DIGITALNI IDENTITETI U OBLAKU

DIPLOMSKI RAD

Dora Klobučar

Osijek, 2022.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Obrazac D1: Obrazac za imenovanje Povjerenstva za diplomski ispit

Osijek, 01.09.2022.

Odboru za završne i diplomske ispite

Imenovanje Povjerenstva za diplomski ispit

Ime i prezime Pristupnika:	Dora Klobučar
Studij, smjer:	Diplomski sveučilišni studij Elektrotehnika, smjer Komunikacije i informatika'
Mat. br. Pristupnika, godina upisa:	D-1291, 13.10.2020.
OIB studenta:	77837013391
Mentor:	Izv. prof. dr. sc. Krešimir Grgić
Sumentor:	,
Sumentor iz tvrtke:	Ljubo Brodarić
Predsjednik Povjerenstva:	Doc. dr. sc. Višnja Križanović
Član Povjerenstva 1:	Izv. prof. dr. sc. Krešimir Grgić
Član Povjerenstva 2:	Mr.sc. Anđelko Lišnjić
Naslov diplomskog rada:	Digitalni identiteti u oblaku
Znanstvena grana diplomskog rada:	**Telekomunikacije i informatika (zn. polje elektrotehnika)**
Zadatak diplomskog rada:	U radu je potrebno analizirati i predstaviti metode upravljanja digitalnim identitetima u oblaku sa osvrtom na najbolje sigurnosne prakse (uz konkretne primjere). Tema rezervirana za: Dora Klobučar Sumentor iz tvrtke: Ljubo Brodarić (Atos)
Prijedlog ocjene pismenog dijela ispita (diplomskog rada):	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 3 bod/boda Postignuti rezultati u odnosu na složenost zadatka: 2 bod/boda Jasnoća pismenog izražavanja: 2 bod/boda Razina samostalnosti: 3 razina
Datum prijedloga ocjene od strane mentora:	01.09.2022.

Potvrda mentora o predaji konačne verzije rada:	*Mentor elektronički potpisao predaju konačne verzije.*
	Datum:

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA
I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

IZJAVA O ORIGINALNOSTI RADA

Osijek, 16.09.2022.

Ime i prezime studenta:	Dora Klobučar
Studij:	Diplomski sveučilišni studij Elektrotehnika, smjer Komunikacije i informatika'
Mat. br. studenta, godina upisa:	D-1291, 13.10.2020.
Turnitin podudaranje [%]:	3

Ovom izjavom izjavljujem da je rad pod nazivom**: Digitalni identiteti u oblaku**

izrađen pod vodstvom mentora Izv. prof. dr. sc. Krešimir Grgić

i sumentora ,

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

Sadržaj

1. UVOD

Računalne tehnologije su značajno napredovale posljednjih nekoliko godina te su prihvaćene od velikog broja organizacija kako bi se smanjili troškovi i omogućio fleksibilan i učinkovit pristup kritičnim podatcima. Brojna poduzeća vide računarstvo u oblaku (*eng. Cloud Computing*) kao platformu za organizacijsku i ekonomsku korist. Računarstvo nudi novi način pristupa računalnim uslugama. Na ovaj način su izložili organizacije rizicima kojih one nisu svjesne. Izazovi u kibernetičkoj sigurnosti su se brzo razvijali kao posljedica razvoja novih tehnologija. Jedno od kritičnih područja na koje treba obratiti pozornost je upravljanje identitetima, točnije digitalnim identitetima u oblaku, gdje višestruki identiteti korisnika mogu djelovati u federalnom okruženju. Postoji kritična potreba za sigurnim pohranjivanjem, dijeljenjem i analiziranjem ogromne količine kompleksnih podataka za određivanje obrazaca i trendova s ciljem poboljšanja kvalitete zdravstvene zaštite, bolje zaštite naroda i istraživanje alternativnih energija [1].

Upravo to pokušava riješiti računarstvo u oblaku, upravljajući ogromnom količinom podataka. Google je predstavio *MapReduce*, okvir koji se koristi za obradu velikih količina podataka na hardveru. *Apacheov Hadoop* distribucijski datotečni sustav se pojavio kao vrhunska komponenta za računarstvo u oblaku u kombinaciji s integriranim dijelovima kao što je primjerice *MapReduce*. Međutim, ni najsuvremeniji računalni sustavi nisu dovoljni zbog činjenice da ne pružaju odgovarajuće sigurnosne mehanizme za zaštitu osjetljivih podataka, budući da oni nemaju sposobnost obrade goleme količine semantičkog web-a i geoprostornih podataka. Kako bi riješili trenutno ograničenje na *Cloud-u,* istraživači su koristili najsuvremenije tehnologije i razvili siguran računalni okvir. Primjerice, moderni dijelovi hardvera za poboljšanje performansi zbog ugradnje dodatne sigurnosne funkcionalnosti, integrirane otvorene dijelove izvornog softvera itd. Neki noviji radovi uključuju ispitivanje *XACML* modela sa *SAML-om* za sigurnost računalnih okvira. Samo upravljanje digitalnim identitetima u oblaku je usko povezano s Web uslugama [1].

Sam diplomski rad se sastoji od 7 poglavlja. Opisan je detaljan pregled različitog upravljanja identitetima kao i sigurno računarstvo u oblaku. U drugom poglavlju su navedene tehnologije upravljanja identitetom. U trećem poglavlju je opisana sigurnost za računarstvo u oblaku te u četvrtom su nešto detaljnije opisani digitalni identiteti u oblaku.

U petom poglavlju će biti više govora o standardima i samim nedostatcima računarstva u oblaku te zaključak .

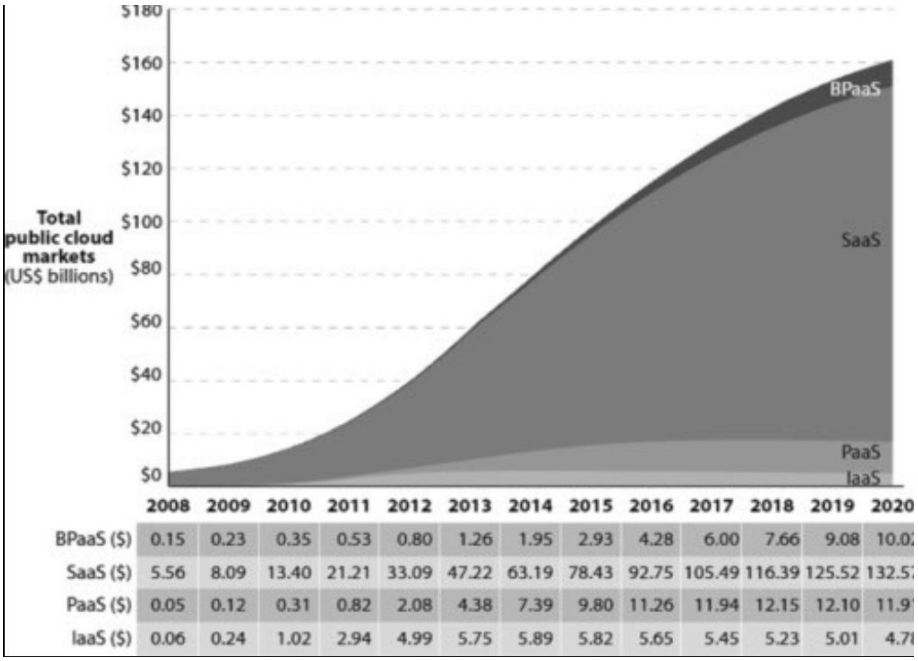
2.KONCEPTI RAČUNARSTVA U OBLAKU

2.1 Definicija računalnog oblaka

Pojam " oblaka (*engl.cloud*) " se koristi u brojnim kontekstima, kao što se koristio i za ATM (*Asynchronous Transfer Mode*) mreže 1990-tih. Međutim, tek je postao popularan 2006 kada je *Eric Schmidt*, izvršni predsjednik *Googlea,* koristio *Cloud* za opisivanje ekonomskog modela povezanog opskrbom Web usluga. Prema [1] računarstvo u oblaku se definira kao model za pristup resursima za izračun i pohranu na zahtjev (kao što je primjerice mrežna infrastruktura, serveri, prostor za pohranu podataka, aplikacije i servisi) dostupan s minimalno napora, u smislu upravljanja uslugama i održavanja.

2.2 Principi koji se koriste u računarstvu u oblaku

Unatoč iznimno velikom porastu samog računarstva na internetu posljednjih nekoliko godina, uz povećan interes medija, osnovni koncept samog oblaka je poprilično čudan. Ovaj poprilično jednostavan koncept uključuje korištenje trećih strana za pohranu podataka i izvođenje računalne obrade. Drugim riječima, računarstvo se sastoji od "iznajmljivanja " opreme od treće strane, koji je poznat kao pružatelj usluga u oblaku. Navedene usluge mogu imati različite i gotovo neograničene oblike, a računalna zajednica redovito uvode nove vrste usluga. Stoga računarstvo u oblaku samo po sebi nije nova ideja. X terminal, popularni 1990-tih su već tada koristili temelje *SaaS-a (engl.Software as Service).* Glavni razvoj posljednjih nekoliko godina je zapravo znatno povećanje opsega samog računarstva. Budući da sada pružatelji usluga u oblaku upravljaju tako velikim količinama podataka, softvera i podataka, morali su biti izgrađeni posebni podatkovni centri. Kao posljedica toga, došlo je do značajnog oslanjanja na tehnologije virtualizacije. *Forrestovo* istraživanje [1] je predviđalo da će globalno tržište računalstva u oblaku porasti sa 40,7 milijardi dolara u 2011. na 241 milijardu dolara do 2020.godine. Predstavljeno istraživanje se može vidjeti na slici 2.1 koja se nalazi u nastavku.



	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020
BPaaS ($)	0.15	0.23	0.35	0.53	0.80	1.26	1.95	2.93	4.28	6.00	7.66	9.08	10.0
SaaS ($)	5.56	8.09	13.40	21.21	33.09	47.22	63.19	78.43	92.75	105.49	116.39	125.52	132.5
PaaS ($)	0.05	0.12	0.31	0.82	2.08	4.38	7.39	9.80	11.26	11.94	12.15	12.10	11.9
IaaS ($)	0.06	0.24	1.02	2.94	4.99	5.75	5.89	5.82	5.65	5.45	5.23	5.01	4.7

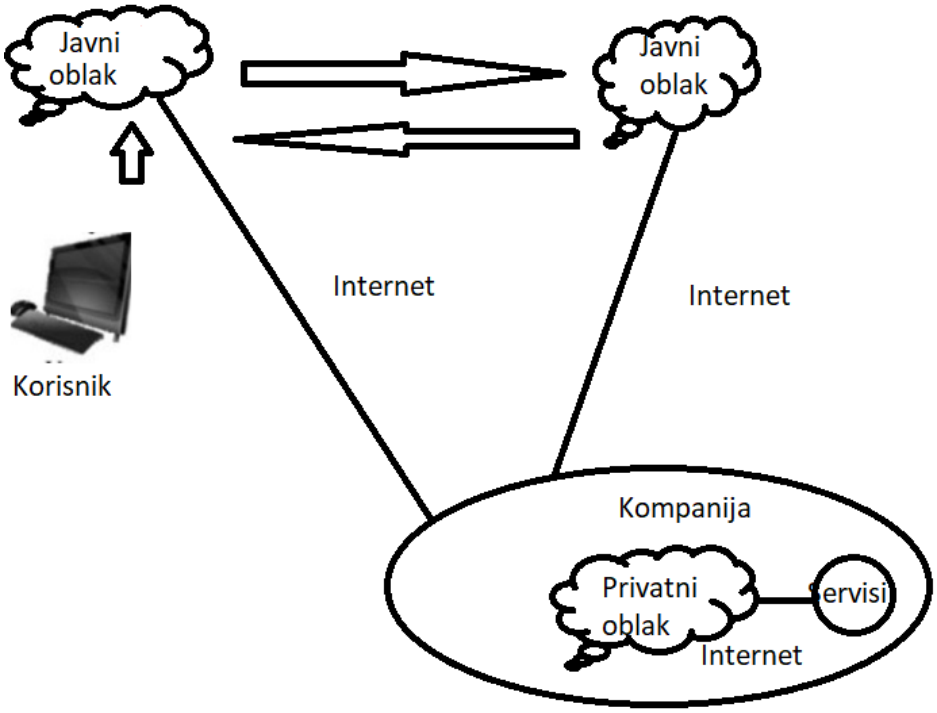
Slika 2.1 *Prognoza tržišta oblaka prema Forrester istraživanju*

Prema najnovijim istraživanjima *Forrest* predviđa da će 2022 godine polovica poslovnih organizacija prihvatiti izvorni *Cloud (slika 2.2)*. To je u skladu s prethodnim trendovima prihvaćanja koji je porastao s 33% 2020. na 42% u 2021. Izvorne tehnologije u Cloud-u obuhvaćaju sve glavne tehnološke domene, poput velikih podataka, umjetne inteligencije ili interneta stvari. Velika je vjerojatnost da će sada tvrtke promijeniti svoju strategiju upravljanja digitalnim identitetima u Cloude-u. Ovo stajalište dijeli i Gartner [1], koji prepoznaje izvorne platforme u Cloude-u kao odgovor na sve veću važnost provođenja digitalnih inicijativa s kojim se susreću brojna poduzeća. Predviđa se da će do 2025. većina izvornih platforma u oblaku poslužiti kao temelj za više od 95% digitalnih inicijativa - porast od 40% u odnosu na 2021. Preporučuje se modernizacija postojećih aplikacija koje mogu imati korist od izvornih tehnologija u oblaku.

Slika 2.2 *Forrestovo predviđanje za 2022.godinu.*

2.3 Modeli implementacije za računalstvo u oblaku

Podaci se mogu eksternalizirati na različite načine. Postoji nekoliko različitih tipova oblaka, bilo privatnih, javnih ili hibridnih [1] kako je i prikazano detaljnije na slici 2.3.



Slika 2.3 *Hibridni oblak*

Privatni oblak, poznat kao interni oblak, dizajniran je za korištenje od strane jedne organizacije i može mu pristupiti samo ta organizacija. Infrastrukturom može upravljati sama tvrtka ili treća strana. On nudi potpunu kontrolu nad infrastrukturom u smislu performansi, pouzdanosti i sigurnosti. No, ovaj tip oblaka se često kritizira da je sličan tradicionalnim vlasničkim firmama poslužitelja, ne nudi sve prednosti poput financijskih ulaganja ili održavanja. *Cloud* zajednice se razlikuju od privatnog oblaka po tome što su proširena na skup organizacija sa zajedničkim ciljevima i misijama. Javni oblak s druge strane, nudi resurse putem pružatelja usluga i otvoren pristup infrastrukturi oblaka svim korisnicima. Nudi niz značajnih prednosti za potrošače, kao što je odsutnost financijskih ulaganja i implikacija održavanja. Međutim, javni oblak ne može jamčiti potpunu kontrolu podataka, mreža i sigurnosti, ograničavajući se na učinkovitost u različitim scenarijima. Hibridni oblaci pak koriste arhitekturu koja kombinira najmanje dvije različite vrste oblaka. Za firmu to podrazumijeva kombiniranje javnog i privatnog oblaka, kako bi se postigla ravnoteža između sigurnosti podataka i usluga koju nudi privatni oblak i dostupnost usluga na zahtjev, korištenjem javnog oblaka kao što je i navedeno na slici 2.3 [1]. Kako bi se riješio problem ograničavanja privatnog i javnog oblaka definira se privatni virtualni oblak (*VPC (engl. Virtual private cloud*)). To je platforma koja se izvršava na vrhu privatnog oblaka. Pruža pružateljima

usluge pristup virtualnoj privatnoj mreži za dizajniranje vlastitih topologija, jamčeći bolje razine sigurnosti od onih koje se susreću u javnom oblaku.

2.4 Svojstva računarstva u oblaku

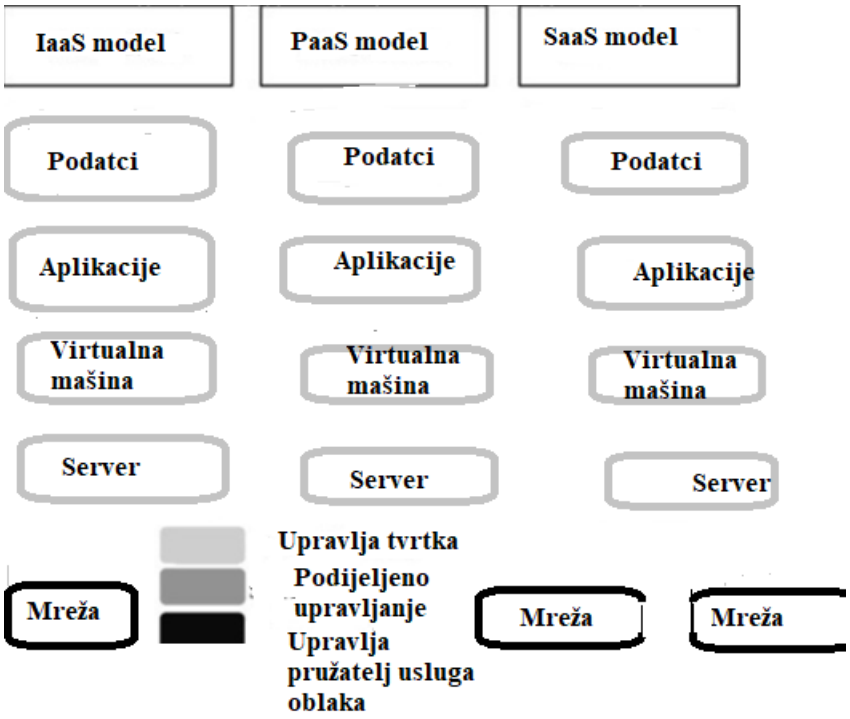
Računarstvo u oblaku mora zadovoljavati pet osnovnih karakteristika prema *NIST-u* (*engl. National Standard of Institute and Technology*) [1]:

1. Samoposluživanje na zahtjev: klijent dobavljača u oblaku bi trebao moći dobiti proširenje ili resurse kao što su kapacitet obrade , mrežna infrastruktura , virtualna mašina itd.…
2. Udruživanje resursa: određena usluga u oblaku trebala bi moći istovremeno posluživati više korisnika, koristeći model s više stanara , s fizičkim i virtualnim resursima dodijeljeni prema zahtjevu kupaca. Kupac ne zna gdje se nalazi lokacija traženih resursa, ali može ugrubo specificirati lokaciju (poput države i grada).
3. Brza elastičnost: sposobnosti oblaka bi trebale biti neograničene korisniku. Navedene sposobnosti bi trebale biti elastično skalabilne i prema van i prema unutra u skladu s potražnjom.
4. Široki pristup mreži: sposobnosti samog oblaka bi trebale biti neograničene korisniku. Navedene sposobnosti bi trebale biti lako skalabilne i prema van i prema unutra u skladu s potražnjom, bez obzira na količinu potrebnih resursa i u bilo koje vrijeme.
5. Mjerene usluge: za svakog korisnika oblaka se mjeri neovisno korištenje resursa, što prije svega omogućuje naplatu uz plaćanje.

2.5 Ostale ključne karakteristike

Odredbe za opremu za eksternalizaciju usluga od strane CPS-a (*engl. Cloud Platform System*) postavljaju pitanja o tome koje se vrste usluga eksternaliziraju i na koji način će klijent koristiti resurse *CPS-a.* Pitanje je tko je zapravo odgovoran za upravljanje različitim vrsta resursa, klijent ili *CPS?* Postoje tri glavna modela usluga u oblaku [1]:

- Softver kao usluga (*Saas (engl.Softwer as Service)):CPS* daje klijentu pristup softverskoj aplikaciji. Sama aplikacija je dostupna korisnicima putem jednostavnog sučelja kao što je ono koje nudi web preglednik. U ovom modelu CPS kontrolira svu temeljnu arhitekturu, uključujući softversku aplikaciju, s iznimkom određenih opcija konfiguracije koje određuju pojedinačni korisnici. Prednost ove usluge je što se ništa ne mora instalirati na korisnikovom lokalnom uređaju, nema nikakve naknade za plaćanje niti potrebnih ažuriranja. Klijentu se naplaćuju samo usluge na zahtjev.
- Platforma kao usluga (*PaaS (platform as Service)*)*:CPS* osigurava platformu, točnije set opreme, kao što je primjerice razvojno okruženje, koji koriste ovlašteni korisnici kako bi razvili svoje vlastite aplikacije. Sam *CPS* nije odgovoran za upravljanje razvijenim aplikacijama, ali zadržava kontrolu osnovnih aplikacija. Korisnik izbjegava potrebu instaliranja razvojnog softvera ili dobivanja materijala, međutim, razvojni tipovi koje sam *Paas* pruža su ograničeni na Web aplikacije te je i raspon jezika također ograničen. Za primjer može se uzeti *Google* koji dozvoljava razvoj koristeći *Java* ili *Pyhton*, dok *Microsoft Azure* predlaže razvoj koristeći .NET.
- Infrastruktura kao usluga (*IaaS (engl. Infrastructure as service)*): *CPS* osigurava sve potrebne računalne resurse, poput procesorske snage, mrežnih sučelja i kapaciteta pohrane, dozvoljavajući ovlaštenim korisnicima razvijanje programa bilo kojeg tipa, od operativnih sustava do aplikacija visokih razina. *CPS* upravlja opremom i izvorima osigurani korisnicima, ali korisnici su odgovorni za sav razvoj koji se provodi pomoću *Cloud* infrastrukture. Prednost navedene infrastrukture je u tome što korisnici mogu pristupiti programima bilo kojeg tipa, bez potrebe za razmatranjem o kontroli, upravljanju i održavanju poslužitelja. Detaljan prikaz koji pokazuje koji entitet firme u *Cloud-u* je odgovoran za različite arhitektonske elemente ovisno o odabranom modelu može se vidjeti na slici 2.4.



Slika 2.4 *Upravljanje arhitekturom prema tipu modela oblaka*

Uz pružatelju usluga u *Cloud-u* , drugi posrednički dobavljači oblaka, poznati kao *Cloud* brokeri, pokušavaju profitirati od tržišta oblaka, to uključuje usluge poput *Dropbox-a*. Oni mogu ponuditi specifičnije modela ako ih smatraju korisnima.

- Identitet kao usluga (*IDaaS (engl.identity as a Service)*): to je infrastruktura za autentifikaciju [1] koja je izgrađena i upravljana pomoću *Cloud* poslužitelja. Temelji se na *SSO* (*engl. Single Sign on*) autentifikaciji koja korisniku omogućuje prijavu s jednim *ID-em* i korisnikovoj autentifikaciji za pristup *Cloud-u.*

2.6 Čimbenici koji utječu na računarstvo u oblaku

Posljednjih godina brojni su čimbenici utjecali na razvoj računarstva u oblaku. *Salesforce.com* je jedan od pionira u žanru, koji su lansirali svoji privatni Cloud 2003. Posljedično ,veliki broj firmi, uključujući svjetske lidere *Google, Microsoft, Amazon i Oracle*, razvili su svoja rješenja za različite modele i usluge, uključujući:

-*SaaS: Google Apps(Gmail, Microsoft Office 365* itd)

-*PaaS:AWS Elastic Beanstalk, Google App Engine itd.*

-IaaS: Amazon EC2,Google Compute Engine itd..

Tablica 1. Primjeri usluga računarstva u oblaku

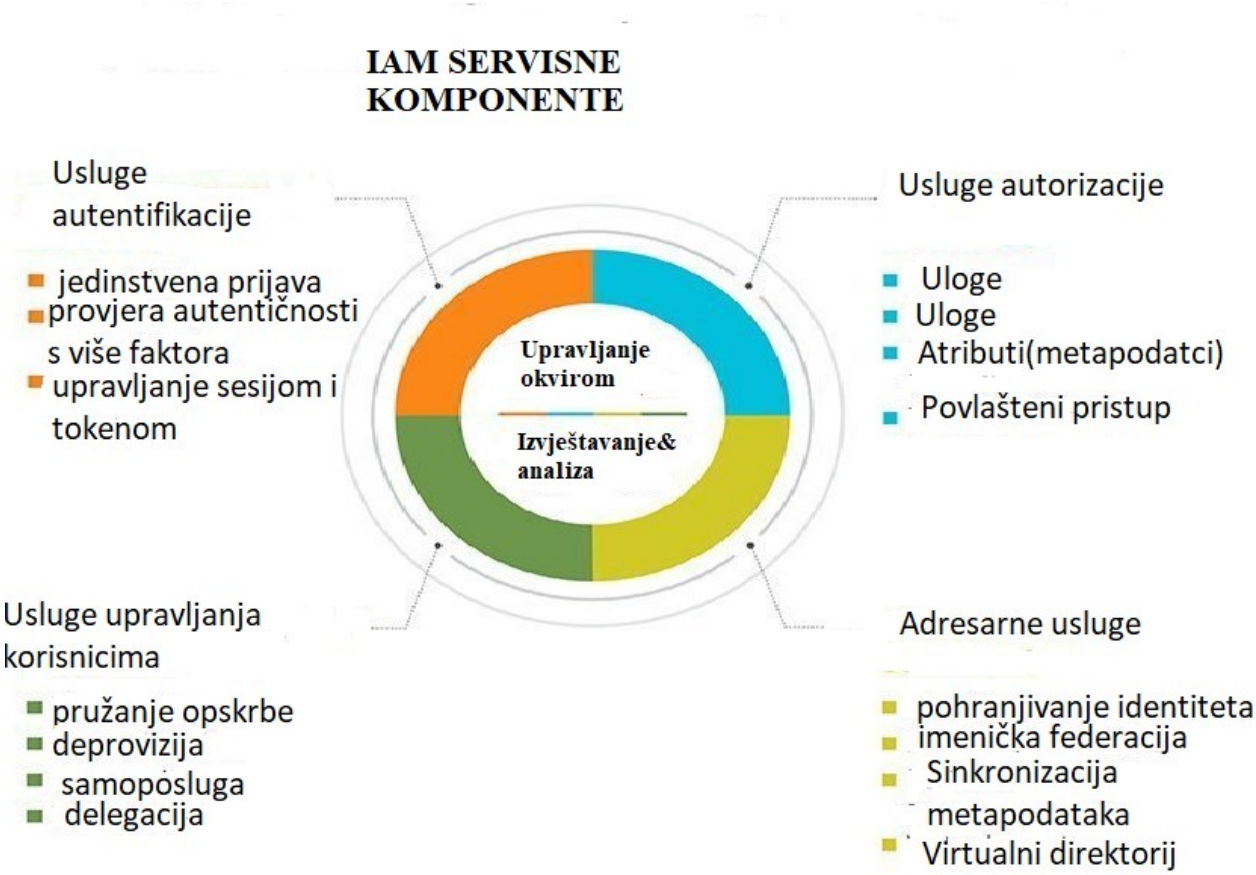
Pružatelji usluga		***Proizvodi***	***Bilješke***
IaaS	*Amazon Web Services*	*Amazon EC2*	
	IBM	*IBM Smart Business Cloud*	
	EMC/Vmware	*Hypervisor vSphere*	
	Microsoft	*Microsoft System Centre*	
	GoGrid	*Cloud*	
		Hosting	
	Eucalyptus Systems	*Eucalyptus(open sources)*	
	RackSpace	*OpenStack(open sources)*	
	Citrix	*CloudStack*	
	Numergy	*OpenStack*	
	CloudWatt	*OpenStack*	
	HP	*OpenStack*	
	DataPipe	*CloudStack*	

PaaS	*Microsoft*	*Microsoft Azure*	*Podržani jezici :Java, PHP, RUBY,ASP.NET*
	Google	*Google App Engine*	*Podržani jezici :Pyhton, Java ,JEE,PHP*
	Salesforce	*Heroku*	*Podržani jezici : Java, Apex, Flex*

Paralelno s ovim razvojem brojnih čimbenika, računarstvo u oblaku je također razvijeno za video igrice. *Cloud gaming* ([1]) se sastoji od izvođenja igara na skupu poslužitelja, a ne putem klijentove konzole i promovirana je od strane *Microsoft-a* i *Sony-a*. Računarstvo u *Cloud-u* je danas temelj mobilnih mreža. Cilj ovakvog pristupa je eksternalizirati lokalne podatke i aplikacije s mobilnog terminala na namjenske poslužitelje. To značajno smanjuje lokalnu potrošnju mobilnih platformi, nešto što je nužno ograničeno, također se smanjuje energetska potrošnja te se produžuje trajanje baterije. Operatori poput *Vodafonea*, *Orange* i *Verizon-a* trenutno idu u tom pravcu. Međutim, nedavni razvoj mobilnog računarstva [1] prelazi prethodno napisane okvire i predlaže modele koji smatra mobilne uređaje pružateljima usluge. U navedenom slučaju, tako zvani virtualni *Cloud* može skupljati podatke s različitih mobilnih uređaja pomoću senzora ili mobilne aplikacije za upravljanje korisnikovim podatcima. Analiziranje prikupljenih podataka i njihovo procesiranje pomoću statističkih metoda ili "rudarenja" podatcima može biti značajno za *Cloud,* omogućujući mu pružanje novih usluga temeljenih na prikupljanju podataka u stvarnom vremenu. Drugi model koji se koristi je *Cloudlets* [1] koji nastoji približiti *Cloud* mobilnom terminalu. Ovaj model se ponaša kao pružatelj usluge mobilnom terminalu i jedino on ima pristup *Cloud* serverima.

2.7 Upravljanje identitetima i pristupom(IAM-IDENTITY ACCESS MANAGEMENT)

Na slici 2.5 se mogu vidjeti servisne komponente od kojih se *IAM* sastoji.



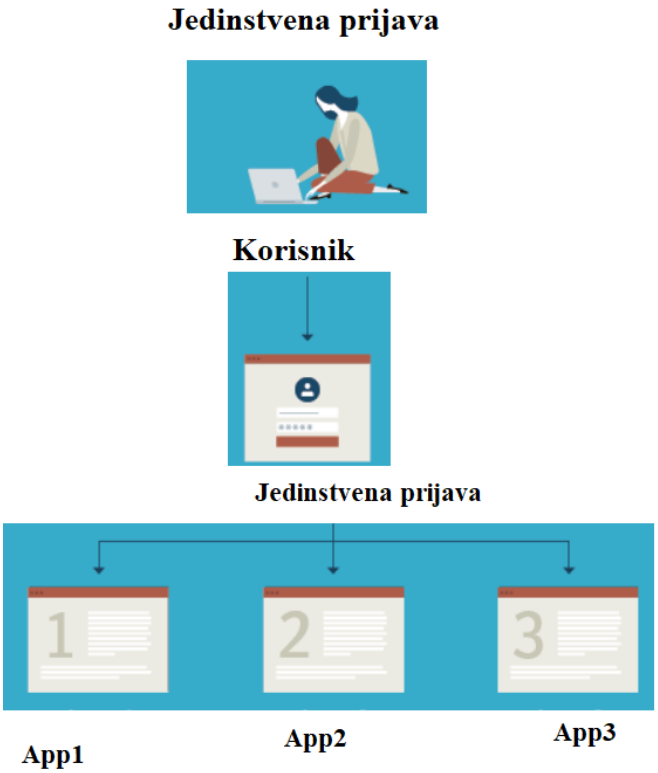
Slika 2.5 *IAM servisne komponente*

Dva temeljna koncepta upravljanja digitalnim identitetom su jedinstvena prijava i federalno upravljanje identitetom. Kao što je i navedeno [1], jedinstven prijava *SSO* je svojstvo korisnika u koje se korisnik prijavljuje jednom i dobiva pristup svim sustavima u federaciji. Na ovaj način se korisnik samo jednom prijavi i ima pristup resursima u federaciji, koaliciji ili organizaciji, a da se pri tomu ne mora ponovno prijaviti na svaki od njih. Dvije vrste *SSO* mehanizma se temelje na *Kerberosovoj* pametnoj kartici. Pomoću *Kerberosovog* mehanizma *TGT (engl.Ticket Granding Ticket)* se koristi za dodjelu vjerodajnica. *ESSO (engl. Enterprise Single Sign On)* pruža podršku za minimiziranje broja lozinki korisničkih id-ova prilikom pristupa više aplikacija. Kao što je ranije navedeno, federativni identitet ili federacija identiteta opisuje standarde, tehnologije i slučajeve upotrebe koji služe kako bi se omogućila prenosivost podataka o identitetu preko inače autonomnih sigurnosnih domena [1]. Slučajevi korištenja uključuju tipične slučajeve upotrebe, uključujući jednostruke prijave na više domena, web-based. Jedan od važnijih koncepata upravljanja digitalnim identitetima su metasustavi. Definira se kao interoperabilna arhitektura koja omogućuje da se digitalni identitet temelji na višestrukoj podlozi tehnologije, implementacije i pružatelja usluge. S ovakvim pristupom, korisnici mogu zadržati svoje identitete i izabrati identitet

sustava koji njima odgovara tako da će upravljati njihovim identiteta prilikom migracije na različite tehnologije. Uloga identiteta meta sustava je pružatelj identiteta, pouzdane strane i subjekti. Oslonjene strane su te koje zahtijevaju identitet kao što su razne usluge. Informacijska kartica se definira kao implementacija mest sustava identiteta, koje ljudi mogu koristiti kod jednokratne prijave budući da se korisnici mogu prijaviti samo na jednom mjestu i imati pristup raznim resursima na web-u. Različite web stranice se sada implementiraju udružene s *Open ID-om*. *Open ID* je decentralizirana identifikacija korisnika, koja korisnicima omogućuje prijavu na brojne usluge s istim digitalnim identitetom. To je ustvari *URL* i korisnik je autentificiran svojim *Open ID-om* pružateljem usluge. Brojne korporacije poput *Microsofta* i *Symantec-a* ga podržavaju. Primjerice, *Microsoft* pruža interoperabilnost između *Open ID-a* i njegovih *Windows CardSpace-a*. *Open ID* proširuje entitete metasustava i sastoji se od krajnjeg korisnika koji želi potvrditi svoj identitet na *Web* mjestu. Identifikator se definira kao *URL* kojeg je izabrao krajnji korisnik. Pružatelj usluge omogućuje usluge registracije *Open ID URL-ova* i pruža *Open ID* provjeru autentičnosti. Pouzdana stranica se definira kao stranica koja želi potvrditi identifikator krajnjeg korisnika [1]. Poslužitelj provjerava identifikator krajnjeg korisnika, dok kod korisničkog agenta korisnici pristupaju pouzdanoj strani ili pružatelju usluge putem samog korisničkog agenta (primjerice: preglednik). Uporaba *Open ID-a* se izvodi tako da korisnik posjećuje web stranicu kako bi zatražio uslugu. Oslanjajuća strana posjeduje *Open ID* obrazac, što je zapravo mjesto gdje se korisnik prijavljuje. Korisnik tada daje svoj prethodno osiguran identitet logičkom procesu. Pomoću navedenih informacija oslanjajuća strana otkriva web stranicu pružatelja identiteta. Lozinka je druga bitna stavka u upravljanju identitetima. Definira se kao kontrolni sustav koji savezima omogućuje zajedničko surađivanje radi dijeljenja web-baziranih sustava. Na taj se način definira protokol za provjeru autentičnosti informacija i korisničkih atributa od izvora do odredišta. Izvorna stranica tada može koristiti atribute za donošenje odluke o korisnikovoj kontroli pristupa. Ovaj srednji sloj softvera koji se temelji na webu koristi *SAML*. U prvoj fazi, izvorna stranica preusmjerava korisnika na njihovu početnu stranicu, dobivajući potvrdu za korisnika koji je autentificiran na početnoj stranici [1].

U drugoj fazi, stranica resursa vraća potvrdu atributu autoriteta matične stranice i vraća skup atributa korisnika na temelju kojih se može pristupiti kontrolnoj odluci. Postoje neki problemi kod jedinstvene prijave lozinkom. Kako izvorna stranica prepoznaje početnu stranicu korisnika? Tako da njome upravlja sustav model povjerenja. Postupak provjere autentičnosti se izvodi na sljedeći način: kada izvorna web stranica zatraži početnu stranicu od korisnika, on ga odabire s popisa pouzdanih stranica koje su već potvrđene certifikatima. Korisnik odabire početnu stranicu sa liste te početna stranica provjerava autentičnost korisnika ukoliko je već registriran. Nakon autentifikacije kućnog poslužitelja, vraća se poruka sa *SAML* oznakom ciljanoj stranici resursa. Stranica resursa (ukoliko se znak podudara) tada daje pseudonim (koji se naziva ručkom (*engl.handle*)) za korisnika i šalje poruku tvrdnje na početnu stranicu kako bi saznao jesu li potrebni atributi dostupni s korisnikove strane. Kako bi se osigurala privatnost, sustav pruža više različitih pseudonima za korisnikov identitet. Kako bi se osiguralo promicanje standarda za promicanje identiteta upravljanja osnovano je društvo *Liberty Alliance*. Razlikujemo *Liberty Identity federation* (federacija identiteta) i *Liberty identity web services*(web usluge identiteta). Federacija identiteta omogućuje korisnicima (primjerice korisnici e-trgovine) provjeru autentičnosti i prijavu na domenu, gdje imaju pristup višestrukim uslugama. Na njemu se temelji SAML 2.0. Standardna web usluga identiteta je otvoreni okvir za implementaciju i upravljanje webom na temelju identiteta usluge. Ove aplikacije uključuju geo-lokaciju, kontakt, poruke i slično. S navedenim uslugama može se upravljati oznakama, blogovima, dijeljenim fotografijama i povezanim društvenim uslugama na webu tako da se očuva privatnost. Privatnost i politika upravljanja ključni su aspekti rada Liberty saveza.

2.7.1 Razlike između SSO-a i FIM-a



Slika 2.6 *Prikaz prijave pomoću SSO-a*

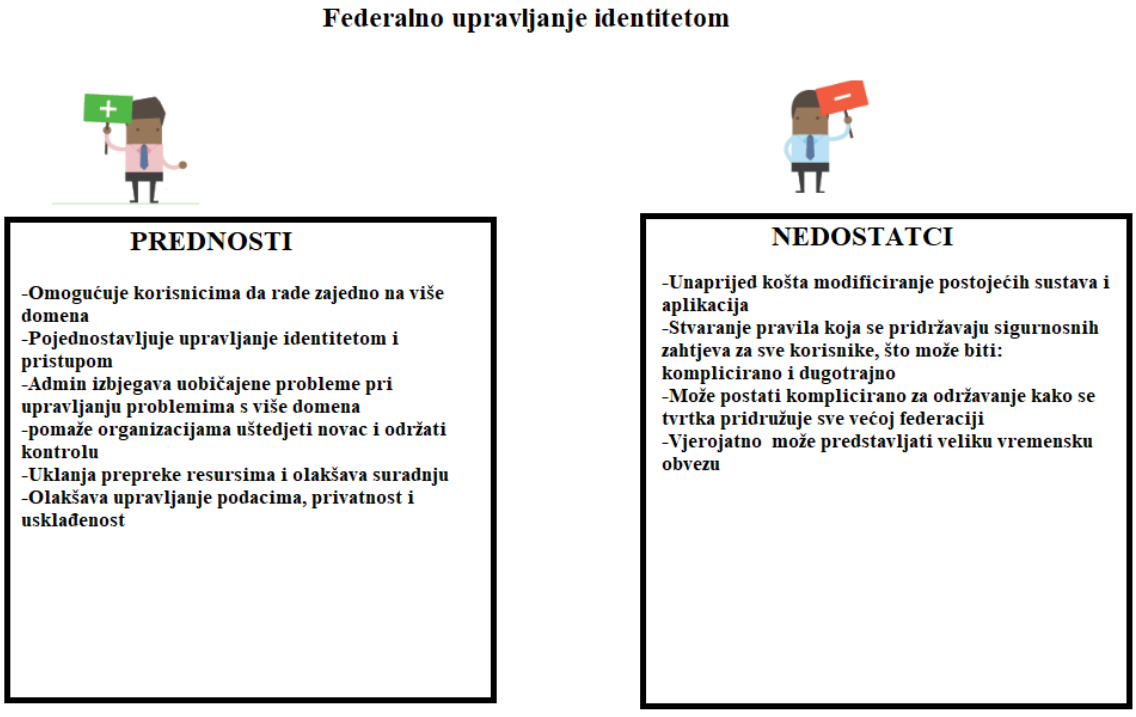
Jedinstvena prijava *SSO* važna je komponenta *FIM-a*, ali nije isto što i *FIM*. Bitno je razlikovati navedena dva pojma. *SSO* omogućuje korisnicima korištenje jednog skupa vjerodajnica za pristup više sustava unutar jedne organizacije. Temelji se na tokenu, što znači da se korisnici identificiraju tokenom, a ne lozinkom.

FIM omogućuje korisnicima pristup sustavima preko federalnih organizacija. Oni mogu koristiti iste vjerodajnice za pristup aplikacijama, programima i mrežama svih članova unutar objedinjene grupe. Omogućuje pristup u jednom koraku višestrukim sustavima u različitim organizacijama. Za razliku od *SSO-a*, *FIM* korisnici ne daju vjerodajnice izravno web aplikaciji, već samom *FIM* sustavu. Organizacije koje implementiraju *SSO* ne koriste nužno *FIM* [1].

2.7.2 Koje su prednosti i nedostaci federalnog upravljanja identitetom?

Kada organizacije rade zajedno na projektu, FIM omogućuje sudionicima pristup i dijeljenje resursa u svim domenama. FIM također pojednostavljuje proces autentifikacije i autorizacije korisnika sustava unutar federacije.

U isto vrijeme, administratori u svakoj organizaciji i dalje kontroliraju razine pristupa u svojim domenama. Oni mogu postaviti dopuštenja i razine pristupa u različitim sustavima u različitim sigurnosnim domenama za korisnika na temelju jednog korisničkog imena. To smanjuje njihov rad i pojednostavljuje upravljanje identitetom i pristupom. Na slici 2.7 se može vidjeti prikaz prednosti i nedostataka upravljanja federalnim identitetom [2].



Slika 2.7 *Prikaz prednosti i nedostataka upravljanja federalnim identitetom*

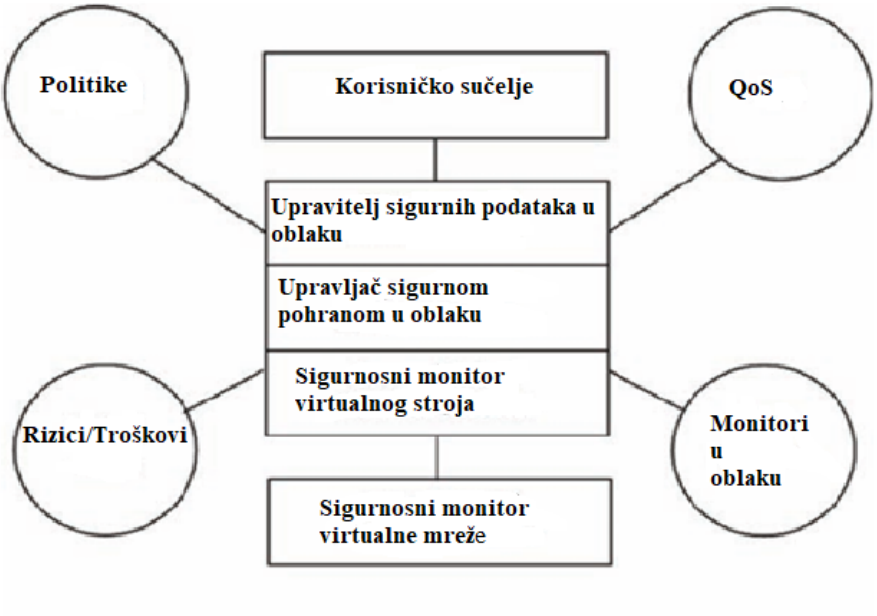
Administratori također mogu izbjeći uobičajene probleme koji se pojavljuju prilikom balansiranja pristupa više domena, kao što je razvoj specifičnog sustava koji olakšava pristup resursima vanjske organizacije. Konsolidacijski pristup FIM-a pomaže organizacijama uštedjeti novac i zadržati kontrolu.

FIM također uklanja prepreke koje često sprječavaju korisnike da jednostavno i sigurno pristupe resursima koji su im potrebni. Omogućuje pogodnost tako da mogu sigurno pristupiti sustavima u različitim domenama bez potrebe da pamte više vjerodajnica ili se više puta prijavljuju. Posljedično, korisnik može uštedjeti vrijeme, minimizirati trenje pristupa i povećati produktivnost. FIM također pojednostavljuje upravljanje podacima, privatnost i usklađenost te smanjuje troškove pohrane. Jedan nedostatak FIM-a su početni troškovi koje organizacije imaju za modificiranje postojećih sustava i aplikacija. To može biti značajan financijski teret za manje organizacije. Drugi izazov je da članovi federacije koji sudjeluju moraju kreirati politike koje se pridržavaju sigurnosnih zahtjeva svih članova. Ovo pregovaranje može biti kompliciran i dugotrajan pothvat kada svako poduzeće postavlja različite zahtjeve i pravila. Naposljetku, organizacija koja sudjeluje može biti članica više od jedne federacije, tako da njezine politike trebaju odražavati pravila i zahtjeve svake federacije. Kako se tvrtka pridružuje dodatnim federacijama, to može postati komplicirano i zahtijevati veliku vremensku obvezu na koju mnoga poduzeća možda nisu spremna [2].

3.SIGURNO RAČUNARSTVO U OBLAKU

Budući da se javlja kritična potreba za sigurnim pohranjivanjem, upravljanjem, dijeljenjem i analizom golemih količina složenih (npr.polustrukturiranih i nestrukturiranih) podataka važno je da postoji određena sigurnost u oblaku. Najveći sigurnosni problem je što vlasnik podataka nema kontrolu nad time gdje se podatci postavljaju. Razlog tomu je što ako netko želi iskoristiti prednosti korištenja računarstva u oblaku, također mora koristiti alokaciju resursa i raspoređivanje koje pruža oblak. Zbog toga se podatci moraju zaštiti usred nepouzdanih procesa. Novi model računarstva pokušava se pozabaviti munjevitim rastom uređaja koji su povezani s internetom te pokušava rukovati ogromnom količinom podataka. Shodno tomu, paradigma računarstva u oblaku [2] dovodi do nekoliko sigurnosnih problema. Primjerice, mapiranje virtualnih strojeva na fizičke strojeve se mora provesti na siguran način. Sigurnost podataka uključuje šifriranje podataka kao i osiguravanje primjene odgovarajućih pravila za dijeljenje podataka. Također, algoritmi za dodjelu resursa i upravljanje memorijom moraju biti sigurni. Tehnike rudarenja mogu biti primjenjive

prilikom otkrivanja zlonamjernog softvera u oblacima. Proširile su se tehnologije razvijene za sigurnu mrežu na siguran oblak. Definiran je slojeviti okvir za sigurno računarstvo u oblaku koji se sastoji od sigurnog sloja virtualnog stroja, sigurnog sloja za pohranu u oblaku, sloj sigurnih podataka u oblaku i sloj za nadzor sigurne virtualne mreže kao što se može i vidjeti na slici 3.1. Međusobne usluge pružaju sloj politike, sloj pouzdanosti i sloj analize rizika.

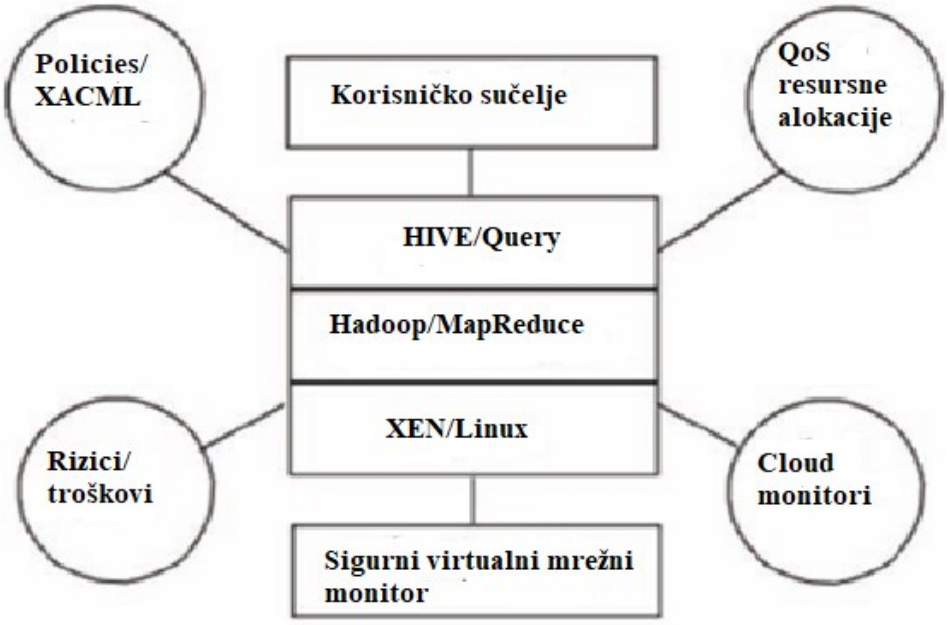


Slika 3.1 *Prikaz slojevitog okvira za siguran oblak*

3.1 Sigurni hipervizori

Za sigurne virtualne uređaje (*VM*) kombiniraju se softverska i hardverska rješenja u virtualnim strojevima za rješavanje problema kao što je *key logger* koji ispituje *XEN* koji je razvijen na sveučilištu *Cambridge* te se istražuju sigurnosti kako bi se zadovoljile potrebe aplikacija (npr. sigurna distribuirana pohrana i upravljanje podatcima). Za sigurnu pohranu na *Cloud-u*, razvijena je infrastruktura za pohranu koja integrira resurse više davatelja kako bi se formirao masivni virtualni prostor za pohranu. Kada se čvor za pohranu ponaša kao domaćin za nekoliko domena, *VM* će se tada stvoriti za svaku domenu kako bi se izolirale informacije i odgovarajuća obrada podataka. Budući da se podatci mogu dinamički kreirati i dodijeliti čvorovima za pohranu,

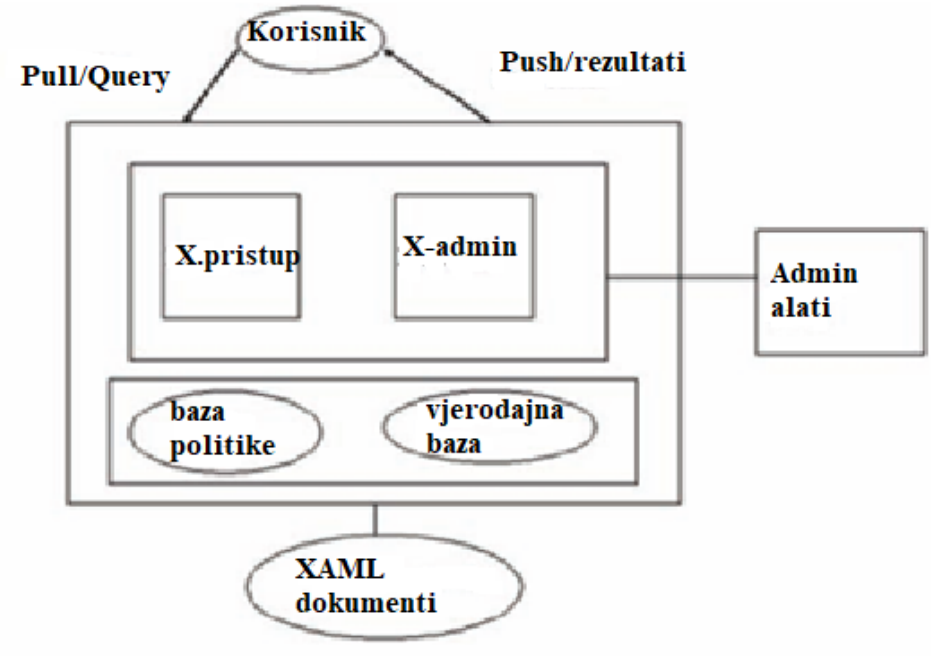
istražuju se sigurne usluge za upravljanje *VM-om*, uključujući upravljanje njegovim spremištem, upravljanje diverzifikacijom *VM-a* te upravljanje kontrolom pristupa. Od tehnologija su korišteni *Hadoop* i *MapReduce*. *MapReduce* je programska paradigma koja omogućuje skalabilnost velikom broju poslužitelja u *Hadoop* klasteru. On je srce *Apache Hadoop-a.* Sam izraz ukazuje na dva različita i odvojena zadatka koja obavljaju *Hadoop* programi .Za sigurno upravljanje podatcima u Cloud-u razvijeni su sigurni algoritmi za obradu upita za *RDF i SQL* podatke u oblacima koji se temelje na *XACML-u* koja koristi *Hadoop/MapReduce framework*. Slika 3.2 ilustrira sve tehnologije koje su korištene za svaki od navedenih slojeva [3].



Slika 3.2 *Slojeviti prikaz tehnologija koji koriste Hadoop/MapReduce framework*

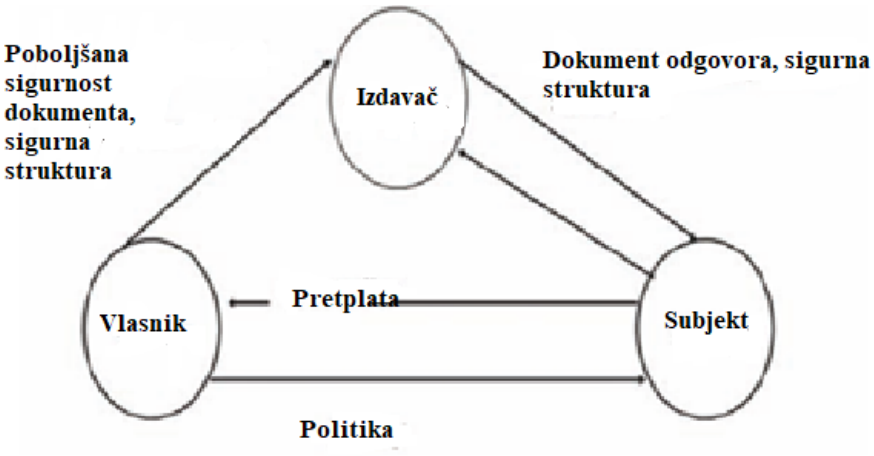
Računarstvo u oblaku olakšava pohranu podataka na udaljenim mjestima kako bi se maksimalno iskoristili resursi. Zbog toga je potrebno zaštiti te podatke i pristup treba biti omogućen samo ovlaštenim osobama. To u biti znači sigurno objavljivanje od treće strane koja je potrebna za prijenos podataka izvana te i vanjske objave. Razvijene su tehnike za objavljivanje podataka od treće strane na siguran način. Pretpostavljeno je da su podatci predstavljeni u obliku *HTML*

dokumenta. To se može i potvrditi tako što je na internetu veliki broj podataka *u HTML-u*. Prema *Bertinu* [3] predložen je okvir za kontrolu, u kojem je sigurnosna politika specificirana ovisno o korisničkim uslugama i vjerodajnicama (prikaz se može vidjeti na slici 3.3). Kako bi pristupili *XML* dokumentu korisnici moraju posjedovati vjerodajnice. Akreditacije ovise o njihovim ulogama. Primjerice, profesor ima pristup svim pojedinostima o studentu, dok tajnik ima pristup samo administrativnim podatcima. *XML* specifikacije se koriste za određivanje sigurnosnih politika. Pristup se odobrava za cijeli *XML* dokument ili za dio dokumenta. Uz određene uvjete, kontrola pristupa se može proširiti i na *XML* stablo. Na primjer, ako je pristup odobren korijenu, to ne znači nužno da je pristup dodijeljen svim podređenima. Može se odobriti pristup XML shemi, ali ne i instancama dokumenata. Također se može dodijeliti pristup i određenim dijelovima dokumenata. Primjerice, profesor nema pristup medicinskim podatcima studenta, ali ima pristup ocjenama studenta i akademskim akreditacijama. Dizajniran je sustav za provedbu politike kontrole pristupa. Glavni cilj je koristiti modificirani oblik pogleda tako da korisnik bude ovlašten koristiti samo ono što je navedeno u pravilima. Potrebno je provesti više istraživanja o kontroli pristupa na temelju uloga za XML i semantičku mrežu. Prema *Bertinu* [3] raspravlja se o sigurnoj objavi XML dokumenta, što se i može vidjeti na slici 3.3. Ideja je imati nepouzdane izdavače treće strane. Vlasnik dokumenta određuje politiku kontrole pristupa subjektima. Vlasnik šalje dokumente nakladniku. Kada subjekt zatraži dokument, izdavač će primijeniti politike koje su značajne za predmet i dati dijelove dokumenata subjektu.



Slika 3.3 *Prikaz okvira kontrole pristupa*

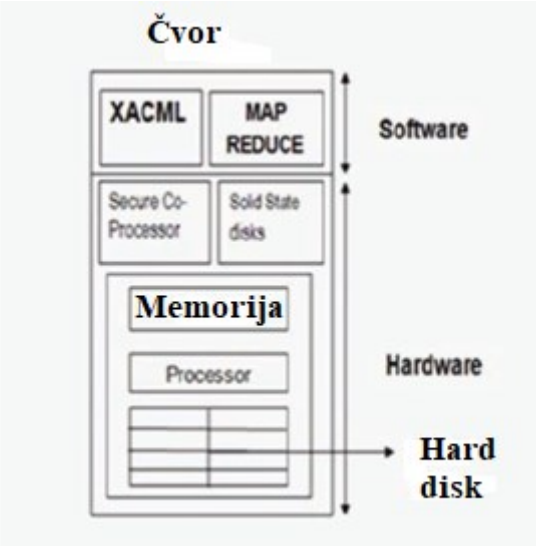
Izdavaču se ne može vjerovati, budući da može dati lažne informacije subjektu. Stoga će vlasnik šifrirati različite kombinacije dokumenata i politika svojim privatnim ključem. Koristeći *Merkle* potpise i tehnike šifriranja, subjekt može provjeriti cjelovitost i autentičnost dokumenata(prikaz sigurnog objavljivanja XML dokumenata se može vidjeti na slici 3.4). U *Cloud* okruženju, izdavač treće strane je stroj koji je pohranio osjetljive podatke na *Cloud.* Navedeni podatci moraju biti zaštićeni i tehnike koje su ranije navedene se moraju primijeniti kako bi se očuvala autentičnost i potpunost.



Slika 3.4 *Sigurno objavljivanje treće strane*

3.2 Šifrirana pohrana podataka za Cloud

Budući da će se podatci na oblaku smjestiti bilo gdje, važno je da podatci budu šifrirani. Korišten je siguran co-procesor kao dio infrastrukture oblaka kako bi bila omogućena učinkovita šifrirana pohrana osjetljivih podataka. Postavlja se jedno od temeljnih pitanja: Zašto ne implementirati softver na hardver koji nudi trenutni sustav računarstva u *Cloud* kao što je i *Open Cirrus?* Uzima se u razmatranje navedena opcija. *Open Cirrus* pruža ograničen pristup na temelju svog ekonomskog modela (*npr.Virtual Cash*). Nadalje, *Open Cirrus* ne pruža potrebnu hardversku podršku (npr. Sigurni co-procesori). Ugrađivanjem sigurnog (*SCP*) u infrastrukturu oblaka, sustav može učinkovito rukovati šifriranim podatcima (slika 3.5).



Slika 3.5 *Dijelovi predloženog instrumenta*

U osnovi SCP je hardver, koji je otporan na neovlašteno korištenje, sposoban za ograničeno računanje opće namjene. Primjerice, *IBM 4758 Crypto Graphic Coprocessor* jedno je pločasto računalo koje se sastoji od *CPU-a*, memorije i kriptografskog hardvera posebne namjene sadržanoj u ljusci otpornoj na neovlašteno korištenje, certificiranoj na razini 4 prema *FIPS PUB 140-1*. Kada je instaliran na poslužitelju, može raditi lokalna računanja koja su potpuno skrivena od poslužitelja. Ako se otkrije neovlašteno korištenje, tada sigurni co-procesor briše internu memoriju. Budući da je siguran co-procesor otporan na neovlašteno korištenje, moglo bi doći u iskušenje da pokrene cijeli poslužitelj za pohranu osjetljivih podataka na sigurnom co-procesoru. Guranje cjelokupne funkcionalnosti pohrane podataka u siguran co-procesor nije izvedivo zbog velikog broja razloga. Prije svega, zbog ljuske otporne na neovlašteno korištenje, sigurni co-procesori obično imaju ograničenu memoriju (samo nekoliko megabajta RAM-a i nekoliko kilobajta nepostojane memorije) i računalne snage. Performanse će se s vremenom poboljšati, ali problemi kao što su rasipanje topline/potrošnja energije(koje se mora kontrolirati kako bi se izbjeglo otkrivanje obrade) stvorit će jaz između opće namjene i sigurnog računalstva [4].

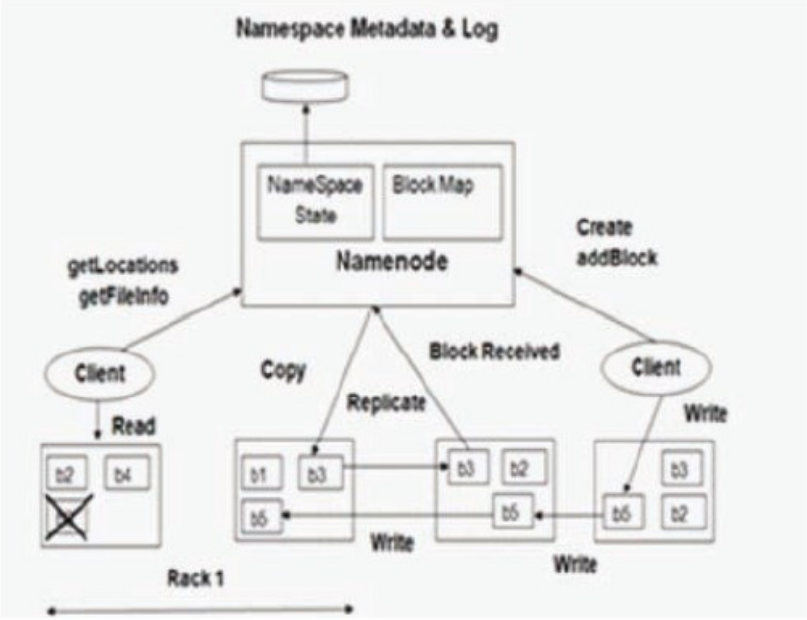
Drugi problem je taj što softver koji radi na *SCP-u* mora biti potpuno pouzdan i provjeren. Navedeni sigurnosni zahtjev podrazumijeva da softver koji radi na *SCP-u* treba biti što jednostavniji. Mogu se šifrirati osjetljivi skupovi podataka pomoću nasumičnih privatnih ključeva i kako bi se ublažio rizik od otkrivanja ključeva, može se koristiti hardver otporan na neovlašteno korištenje za pohranu nekih od ključeva za šifriranje/dešifriranje (tj. Glavni ključ koji šifrira sve ostale ključeve). Budući da se ključevi, ni u jednom trenutku neće nalaziti u memoriji nešifrirani,

napadač ne može memorizirati ključeve tako što će napraviti snimku sustava. Također, svaki pokušaj napadača da preuzme kontrolu nad (ili manipulira) su-procesorom, bilo putem softvera ili fizički, očistit će Co-procesor, čime će eliminirati način dešifriranja svih osjetljivih informacija. Ovaj okvir će olakšati sigurnu pohranu podataka i osiguranu razmjenu informacija. Na primjer, *SCP-ovi* se mogu koristiti za integraciju podataka za očuvanje privatnosti, što je važno za sigurno dijeljenje informacija. Provedena su istraživanja na šifriranim podatcima s upitima, kao i sigurnom višedijelnom računanju (*engl.SCM*). Koristeći *SCM* protokol, svatko ima pristup svojim podatcima, ali ne i podatcima partnera jer su oni šifrirani. No, operacije se mogu izvoditi nad šifriranim podatcima, a rezultati operacija dostupni su svima, recimo, u koaliciji. Jedan nedostatak *SCM-a* su visoki troškovi računanja. Međutim, istražuju se učinkovitiji načine za razvoj *SCM* algoritma te kako se ti mehanizmi mogu primijeniti na oblak.

3.3 Sigurna obrada upita koristeći *Hadoop*

3.3.1 Kratki pregled *Hadoop-a*

Glavni dio sustava je *HDFS* koji je distribuirani datotečni sustav baziran na *Java* programskom jeziku. Sadrži kapacitet za rukovanjem velikim brojem čvorova koji pohranjuju petabajte(*PB*) podataka. Idealna veličina datoteke je ona od 64 *MB*. Pouzdanost se postiže repliciranjem na nekoliko hostova. Zadana vrijednost replikacije je 3(tj., podatci se pohranjuju na tri čvora). Dva čvora se nalaze na istom stalku, dok se treći na posebnom. Datotečni sustav tvore grupe podatkovnih čvorova. Podatkovni čvorovi međusobno komuniciraju kako bi regulirali ,prenijeli i replicirali podatke. HDFS arhitektura se temelji na *master/slave* pristupu(slika 3.6). Master čvor je mjesto imenovanja čvorova te sadrži meta podatke. Sadrži stablo direktorija u kojem su sadržani podatci svih datoteka i prati koji su podatci dostupni s kojeg čvora u klasteru. Navedene informacije se pohranjuju kao slike u memoriji. Glavni čvor za imenovanje ostalih čvorova predstavlja kritičnu točku jer sadrži meta podatke. Stoga postoji zamjenski čvor za taj glavni čvor koji se može postaviti na bilo koji stroj. Klijent pristupa čvoru kako bi dobio meta podatke potrebne datoteke. Nakon što dobije meta podatke, klijent izravno razgovara s odgovarajućim datotečnim čvorovima kako bi dobio podatke i ili izvršio IO radnje(Hadoop-a). Na vrhu datotečnih sustava postoji mehanizam map/reduce. Ovaj stroj se sastoji od *Job Trackera.* Klijentske aplikacije ovom stroju šalju poslove mapiranja/smanjivanja. Job Tracker pokušava rad smjestiti u blizini podataka, gurajući rad na dostupne čvorove *Task Tracera* u klasteru [5].



Slika 3.6 *Hadoop distribuirani datotečni sustav*

3.3.2 Neadekvatnosti Hadoop-a

Navedeni sustavi koji koriste *Hadoop-a* imaju sljedeća ograničenja [5]:

1. Nema mogućnost za rukovanje šifriranim osjetljivim podatcima: osjetljivi podatci u rasponu od medicinske dokumentacije do transakcija kreditnim karticama moraju se pohraniti korištenjem tehnika šifriranja radi dodatne zaštite. Trenutno HDFS ne obavlja sigurnu i učinkovitu obradu upita preko šifriranih podataka.

2. Upravljanje semantičkim web podacima: postoji potreba za održivim rješenjima za poboljšanje performansi i skalabilnosti upita prema semantičkim web podacima kao što je *RDF (Resource Description Framework*). Broj RDF skupova podataka raste. Problem pohranjivanja milijardi RDF trojki i mogućnosti njihovog učinkovitog upita tek se treba riješiti .Trenutno ne postoji podrška za pohranjivanje i dohvaćanje RDF podataka u HDFS.

3. Nema detaljne kontrole pristupa: HDFS ne pruža detaljnu kontrolu pristupa. Postoji određeni posao na osiguravanju popisa za kontrolu pristupa za HDFS .Za veliki broj aplikacija, kao što je osigurano dijeljenje informacija, popisi kontrole pristupa nisu dovoljni i postoji potreba za podrškom složenijih politika.

4. Nema jake provjere autentičnosti: korisnik koji se može spojiti na *JobTracker* može poslati bilo koji posao s privilegijama računa koji se koristi za postavljanje *HDFS-a.* Buduće verzije *HDFS-a* podržavat će protokole mrežne provjere autentičnosti kao što je *Kerberos* za provjeru autentičnosti korisnika i enkripciju prijenosa podataka. Međutim, za neke sigurnosne scenarije dijeljenja informacija trebat će nam infrastrukture javnog ključa (*engl.PKI*) za pružanje podrške za digitalni potpis.

3.3.3 Dizajn sustava

Dok s jedne strane postoje sigurni koprocesori koji pružaju hardversku podršku za upite i pohranu podataka, javlja se potreba za razvitkom softverskog sustava za pohranu, ispitivanje i rudarenje podataka. Sve više i više aplikacija u današnje vrijeme koristi semantičke web podatke kao što su *XML* i *RDF* zbog svoje moći predstavljanja, posebno za upravljanje web podacima. Stoga istražujemo načine za sigurno postavljanje upita semantičkih web podataka kao što su *RDF* podaci u oblaku. Korišteno je nekoliko softverskih alata koji su dostupni na tržištu kako bi pomogli u procesu, uključuju sljedeće stavke:

- **Jenna:** Jena je okvir koji je naširoko korišten za rješavanje SPARQL upita preko RDF podataka (Jena). Ali glavni problem s Jenom je skalabilnost. Skalira se proporcionalno veličini glavne memorije. Nema distribuiranu obradu. Međutim, Jena će biti korištena u početnim fazama koraka pred obrade.

- **Pelet:** Koristimo *Pellet* za razmišljanje u različitim fazama. Razmišljamo o upitima u stvarnom vremenu koristeći biblioteke peleta (*Pellet*) zajedno s *Hadoopovim* funkcijama za smanjenje mapa.

- **Pig Latina:***Pig Latin* je skriptni jezik koji radi na vrhu *Hadoopa* ([6]) .Pig je platforma za analizu velikih skupova podataka. Svinjski jezik, *Pig Latin*, olakšava slijed transformacija podataka kao što je spajanje skupova podataka, njihovo filtriranje i primjena funkcija na zapise ili grupe zapisa. Dolazi s brojnim ugrađenim funkcijama, ali također možemo stvoriti vlastite korisnički definirane funkcije za obradu posebne namjene. Koristeći ovaj skriptni jezik, izbjegava se pisanje vlastitog koda za reduciranje karte; oslonac pruža Pig

Latinova skriptarska snagu koja će automatski generirati kod skripte za smanjenje koda na karti.

- **Mahout, Hama:** Ovo su paketi otvorenog koda za rudarenje podataka i strojno učenje koji već povećavaju Hadoop [6]. Korišteni pristup se sastoji se od sigurne obrade *SPARQL* upita preko *Hadoopa*. *SPARQL* je jezik upita koji se koristi za upite *RDF* podataka (*W3C, SPARQL*). Softverski dio koji je korišten je okvir za upite RDF podataka distribuiranih preko Hadoopa[6]. Postoji nekoliko koraka za prethodnu obradu i upit RDF podataka (slika 3.7). S ovim predloženim dijelom, istraživači mogu dobiti rezultate za optimizaciju obrade upita golemih količina podataka. U nastavku će biti raspravljeno o slijedećim koracima razvoja:

- **Prethodna obrada**: Općenito, *RDF* podaci su u *XML* formatu .Kako bi *SPARQL* upit bio uspješno izvršen , predloženi su neki od koraka za prethodnu obradu podataka i izvode se na način da se spremaju prethodno obrađene podatci u HDFS. Korišten je modul N-trostrukog pretvarača koji pretvara RDF/XML format podataka u N-trostruki format budući da je ovaj format razumljiviji. Koristit će se *Jena* okvir kao što je ranije navedeno, za ovu svrhu pretvorbe. U modulu za razdvajanje datoteka na temelju predikata, dijele se sve datoteke N-trostrukog formata na temelju predikata. Stoga je ukupan broj datoteka za skup podataka jednak broju predikata u ontologiji/taksonomiji. U posljednjem modulu koraka pred obrade, dalje se dijele predikatne datoteke na temelju tipa objekta koji sadrži. Dakle, sada svaka datoteka predikata ima određene vrste objekata u sebi. To se radi uz pomoć *Pellet* biblioteke. Prethodno obrađeni podaci pohranjuju se u *Hadoop*.

- **Izvršenje i optimizacija upita:** Razvijen je *SPARQL* modul za izvršenje i optimizaciju upita za *Hadoop*. Budući da se strategija pohrane temelji na podjelama predikata, prvo će se pogledati predikate prisutne u upitu. Drugo, umjesto da se gledaju sve ulazne datoteke, pogledat će se podskup ulaznih datoteka koje su uparene s predikatima. Treće, *SPARQL* upiti općenito imaju veliki broj spajanja u sebi i sva ta spajanja možda neće biti moguće izvesti u jednom *Hadoop* poslu. U sklopu optimizacije troškova, smanjuje se vrijeme obrade upita. Primjer "pohlepne" strategije je pokrivanje maksimalnog broja spajanja u jednom poslu. Za troškovni model spajanje koje se treba prvi izvesti se temelji na zbirnoj statistici (npr. Faktor selektivnosti ograničene varijable, pridruženi trostruki faktor selektivnosti za tri trostruka uzorka, npr. razmotren je skup podatak *LUBM*, recimo trebaju se navesti sve osobe na određenom sveučilištu itd) [6].

U *SPARQL-u:*

```
PREFIX rdf: <http://www.
w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ub: <http://www.lehigh.
edu/~zhp2/2004/0401/univ-bench.owl#>
SELECT ?X WHERE {
?X rdf:type ub:Person .
<http://www.University0.edu>
ub:hasAlumnus ?X }
```

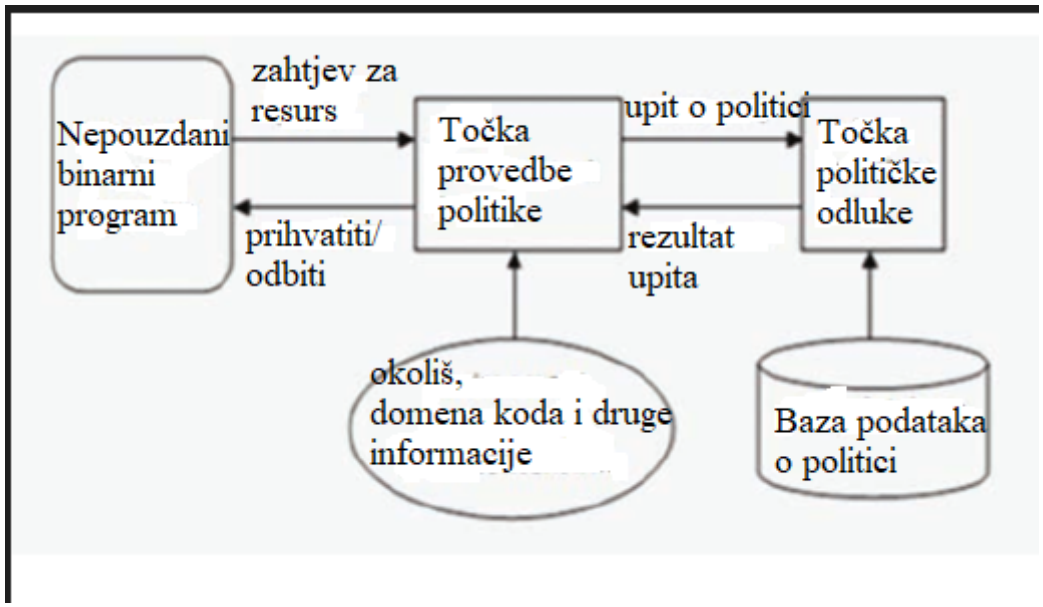
Slika 3.8 *SPARQL*

Optimizator upita će uzeti ovaj unos upita i odlučiti o podskupu ulaznih datoteka koje će pogledati na temelju predikata koji se pojavljuju u upitu. Ontologija i *pellet reasoner* će identificirati tri ulazne datoteke (*underGraduateDegreeFrom, masterDegreeFrom* i *DoctoraldegreeFrom*) povezane s predikatom, "*hasAlumns*". Zatim iz datoteke filtriramo sve zapise čiji su objekti potklasa *Person* pomoću biblioteke peleta. Iz ove tri ulazne datoteke (*underGraduateDe-greeFrom, masterDegreeFrom i Doctoral-degreeFrom*) optimizator filtrira trojke na temelju *<http://www.University0.edu>* kako je potrebno u upitu. Konačno, optimizator određuje zahtjev za jednim poslom za ovu vrstu upita, a zatim se spajanje provodi na varijabli X u tom poslu. S obzirom na sigurnu obradu upita, istražena su dva pristupa. Jedan je prepisivanje upita na takav način da se politike provode na odgovarajući način. Druga je izmjena upita gdje se pravila koriste u klauzuli "gdje" za izmjenu upita [6].

3.3.4 Integracija SUN XACML implementacije u HDFS-u

Hadoop implementacije koje se trenutno koriste provode vrlo grubu politiku kontrole pristupa koja dopušta ili odbija glavni pristup u biti svim resursima sustava kao skupini bez razlikovanja među resursima. Na primjer, korisnici kojima je odobren pristup ime čvoru mogu izvršiti bilo koji program na bilo kojem klijentskom računalu, a svi klijentski strojevi imaju pristup za čitanje i pisanje svim datotekama pohranjenim na svim klijentima. Takva gruba sigurnost očito je neprihvatljiva kada su podaci, upiti i resursi sustava koji ih

implementiraju sigurnosni relevantni i kada se svim korisnicima i procesima ne vjeruje u potpunosti. Trenutni rad [3] to rješava implementacijom standardnih popisa kontrole pristupa za *Hadoop* kako bi se ograničio pristup određenim resursima sustava, kao što su datoteke; međutim, ovaj pristup ima ograničenje da je prisilna sigurnosna politika uklopljena u operativni sustav i stoga se ne može lako promijeniti bez modifikacije operativnog sustava. Provedena je fleksibilnija i detaljnija politike kontrole pristupa na *Hadoopu* tako što dizajniramo implementaciju *Sun XACML-a In-lined Reference Monitor. XACML* [3] je OASIS standard za izražavanje bogatog jezika politika kontrole pristupa u *XML-u*. Subjekti, objekti, odnosi i konteksti su generički i proširivi u *XACML-u,* što ga čini vrlo prikladnim za distribuirano okruženje u kojem mnogobrojne različite potpolitike mogu međusobno djelovati kako bi oblikovala veće, složene politike na razini sustava. Apstraktni mehanizam provođenja *XACML-a* prikazan je na slici 3.9. Nepouzdani procesi u okviru pristupaju resursima relevantnim za sigurnost podnošenjem zahtjeva točki za provedbu politike *(engl.PEP*) resursa. *PEP* preformulira zahtjev kao upit o politici i podnosi ga točki odlučivanja o politici (*PDP). PDP* konzultira sve politike vezane uz zahtjev za odgovor na upit. *PEP* ili odobrava ili odbija zahtjev za resursima na temelju odgovora koji dobije. Dok se *PEP* i *PDP* komponente mehanizma ovrhe tradicionalno provode na razini operativnog sustava ili kao pouzdane knjižnice sustava, predlaže se postizanje veće fleksibilnosti implementacijom u sustav ugrađenih referentnih monitora (*IRM*) [3].



Slika 3.9 *Arhitektura provođenja XACML-a*

IRM provodi sigurnosne provjere tijekom izvođenja tako da te provjere ugradi izravno u binarni kod nepouzdanih procesa. Ima prednost što se politika može primijeniti bez modificiranja operacijskog sustava ili knjižnica sustava. *IRM* politike mogu dodatno ograničiti programske operacije koje bi moglo biti teško ili nemoguće presresti na razini operacijskog sustava. Na primjer, dodjele memorije u *Javi* se implementiraju kao *Java bytecode* instrukcije koje ne pozivaju nikakav vanjski program ili knjižnicu. Provođenje fino-zrnate politike vezane za memoriju kao tradicionalnog referentnog monitora u *Javi* stoga zahtijeva modificiranje *Java* virtualnog stroja ili *JIT*-kompilatora. Nasuprot tome, *IRM* može identificirati ove sigurnosno relevantne upute i umetnuti odgovarajuće zaštitne mjere izravno u nepouzdani kod za provođenje politike. Konačno, *IRM* može učinkovito provoditi sigurnosne politike temeljene na povijesti, a ne samo politike koje ograničavaju pojedinačne sigurnosne politike događaji. Korišten je *IRM* za provođenje pravila pravednosti koja zahtijevaju da nepouzdane aplikacije dijele onoliko podataka koliko zatraže. To sprječava procese da utječu na napade uskraćivanja usluge na temelju ponašanja slobodnog učitavanja. Kod koji je *IRM* ubacio u nepouzdanu binarnu datoteku ograničava svaku operaciju programa temeljenu na prošloj povijesti programskih operacija, a ne izolirano. To uključuje ubacivanje varijabli stanja sigurnosti i brojača u nepouzdani kod, što je teško učinkovito postići na razini operacijskog sustava (slika 3.9). Jezgra IRM okvira sastoji se od binarnog *rewritera*, koji statički modificira binarni kod svakog nepouzdanog procesa prije nego što se izvrši kako bi ubacio zaštitare oko potencijalno opasnih operacija. Predložena implementacija binarnog prepisivanja temeljit će se na *SPoX-u* (*XML* sigurnosne politike) [3] koji je razvijen za provedbu deklarativnih, *XML-based, IRM* politika za *Java bytecode* programe. Kako bi bila pružena snažna sigurnosna jamstva opisanog sustava , primijenit će se tehnologije automatizirane provjere softvera, uključujući provjeru tipa i modela, koje su prethodno korištene za certificiranje izlaza binarnih prepisivača [3]. Takva certifikacija omogućuje malom, pouzdanom verifikatoru da neovisno dokaže da prepisani binarni kod zadovoljava izvornu sigurnosnu politiku, čime se pomiče relativno veći binarni prepisivač iz pouzdane računalne baze sustava.

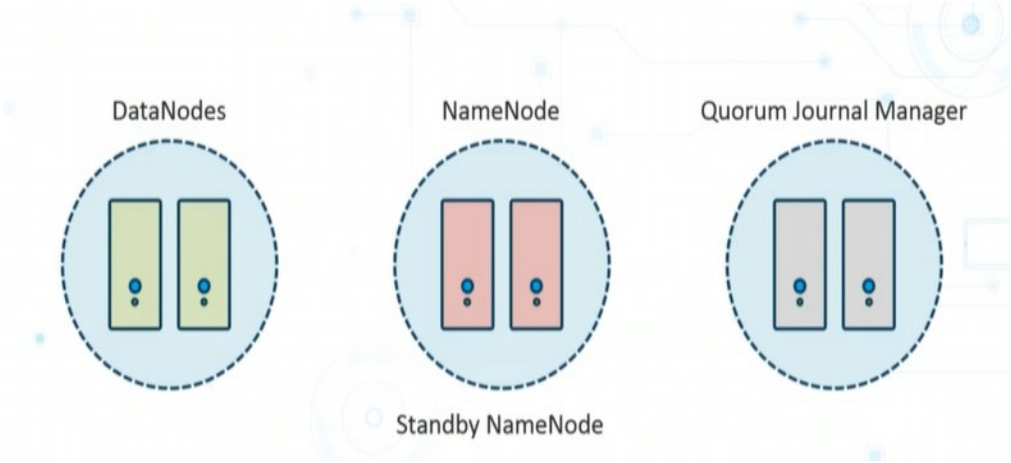
3.3.5 Jaka autentifikacija

Trenutno Hadoop ne provjerava autentičnost korisnika. To otežava provedbu kontrole pristupa ili sigurnosno osjetljive aplikacije i olakšava zlonamjernim korisnicima da zaobiđu provjeru dopuštenja datoteke koju obavlja HDFS. Kako bi riješila ove probleme, zajednica otvorenog koda aktivno radi na integraciji Kerberos protokola s Hadoop-om [5]. Povrh predloženog protokola Kerboros, za neke aplikacije sa sigurnim informacijama može postojati potreba za dodavanjem jednostavnih protokola za provjeru autentičnosti sa sigurnim koprocesorima. Iz tog razloga korištenom sustavu se može dodati jednostavnu infrastrukturu javnih ključeva tako da se korisnici mogu samostalno autentificirati sa sigurnim koprocesorima kako bi dohvatili tajne ključeve koji se koriste za šifriranje osjetljivih podataka. Može se koristiti infrastruktura javnog ključa otvorenog koda kao što je implementacija OpenCA PKI (OpenCA) [5].

3.3.6 Apache Zookeper za HDFS

Uz četiri bitne komponente Hadoopa koje uključuju Yarn, HDFS ,MapReduce potrebno je spomenuti i četvrtu komponentu Zookeper koja nije striktno dio Hadoopa ali sadrži [7] neke pakete u najnovijim verzijama koji mogu pokretati Hadoop u višim verzijama. Što je točno Zookeper? Alat održavan od strane Apache softvera i koristi se za održavanje koordinacije i sinkronizacije u distribuiranom sustavu. Također pruža veliki broj usluga koje su povezane s distribuiranim sustavima, to uključuje imenovanje servisa, konfiguracijski menadžment i tako dalje. Još jedna stvar koju omogućuje je i na kojoj je fokus je da omogućava Hadoop komponentama da djeluju u načinu visoke dostupnosti. Zookeper omogućuje čvor visoke dostupnosti usluga ime čvor servisa. Čvor visoke dostupnosti odnosi se na činjenicu da ako se usluge čvora pokvare, tada se postavlja standby ili rezervni sustav koji zauzima njegov prostor. Sve bi to trebalo biti transparentno za korisnike sustava te oni ne bi trebali biti svjesni činjenice da je došlo do bilo kakvog neuspjeha. S njihove točke gledišta, usluga je uvijek dostupna ili vrlo dostupna. Dakle, jedan od načina na koji Zookeper omogućuje da usluga ime čvor bude visoko dostupna je jednostavno automatiziranje procesa nadilaženja greške. Dakle, ako bi glavna usluga ime čvor propala, Zookeeper dopušta sigurnosnoj kopiji da zauzme svoje mjesto neprimjetno, bez ikakve ručne intervencije. Kada se raspravlja o HDFS-u u, pokriva se činjenica da u klasteru postoji više podatkovnih čvorova. A kada se HDFS izvodi u čvoru visoke dostupnosti, svatko ima svoju uslugu ime čvor koja zapravo ima primarni i rezervni ime

čvor (engl.NameNode). Ako primarni NameNode iz nekog razloga ne uspije, tada će čvor biti u stanju pripravnosti i sve će to biti besprijekorno. Jedan od uvjeta za nesmetano prebacivanje je da oba ova NameNode-a trebaju imati pristup istom datotečnom sustavu. A to je nešto što se postiže korištenjem Quorum Journal Managera. A to je zapravo preferirani način u Hadoopu za postizanje visoke dostupnosti. Dakle, s obzirom na ovu arhitekturu, gdje se točno Zookeeper uklapa? Pa, jedna stvar koju treba izbjegavati je da i u primarnom NameNodeu i u pripravnom NameNodeu iz nekog razloga mislimo da su oba primarna. A zatim istovremeno izvršiti operacije pisanja na podatkovnim čvorovima. Na slici 3.10 možemo vidjeti prikaz opisane arhitekture [7].



Slika 3.10 *Arhitektura Zookepera*

NameNode i *Standby NameNode* povezani su s oba *DataNode*.To će sigurno ostaviti datotečni sustav u nedosljednom stanju, a to će biti situacija iz koje će se biti gotovo nemoguće oporaviti. Svaki od ovih čvorova imena također će ažurirati *Quorum Journal Manager*, što će zauzvrat dovesti do daljnjih komplikacija. Ovaj problem se naziva scenarij podijeljenog mozga i to je čest problem u distribuiranim sustavima koji treba izbjegavati. Dakle, jasno je da postoji potreba za sinkronizacijom između svakog od čvorova imena, primarnog i pripravnog. Mora biti potpuno jasno koji je primarni u bilo kojem trenutku.

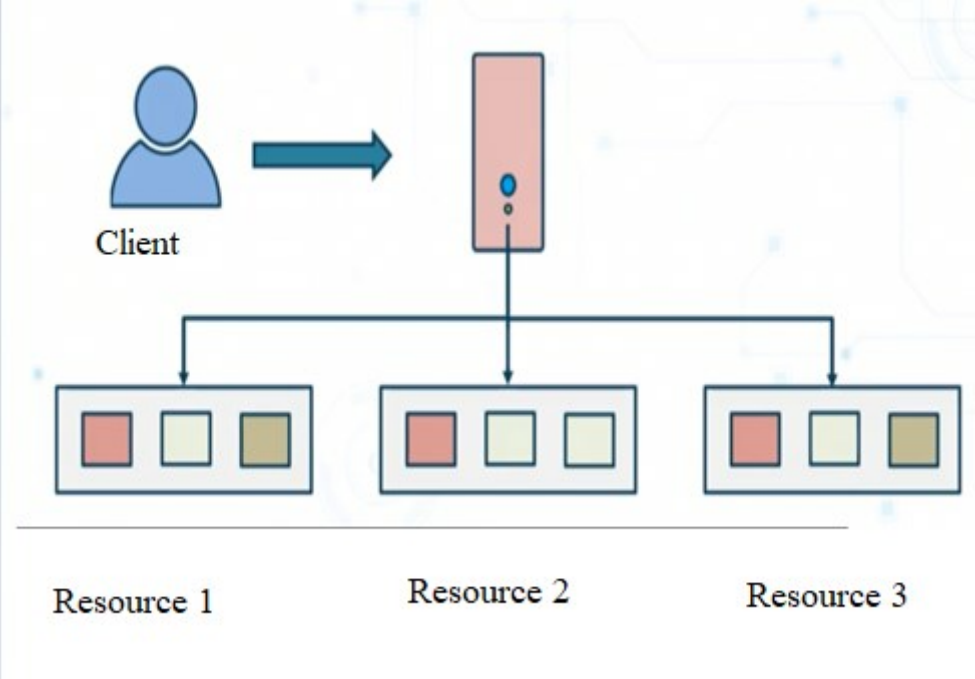
I to je točno mjesto gdje se *Zookeeper* uklapa. Postojat će određeni broj Zookeeper čvorova obično neparan broj, tako da je lakše postići većinu. Dakle, potrebno je ispitati njegovu ulogu malo detaljnije. Tako je viđeno da je u starijim verzijama Hadoopa, u verziji 1.0, na primjer, usluga NameNode predstavljala jednu točku kvara. Ako bi se stroj koji sadrži *NameNode* iz nekog razloga srušio, tada bi se izgubila sva preslikavanja pojedinačnih

blokova datoteka distribuiranih u klasteru. Jednostavno rečeno, neće biti moguće locirati nijednu datoteku na HDFS-u ako se usluga *NameNode* pokvari. Rješenje je, naravno, da HDFS radi u načinu visoke dostupnosti, gdje postoji primarni *NameNode* kao i rezervni *NameNode*. Međutim, u ovom slučaju mora postojati neka vrsta sinkronizacije između oba poslužitelja *NameNode* kako bi znali koji je zapravo primarni u bilo kojem trenutku. To se može postići korištenjem brojnih čvorova *Zookeeper-a*. Od toga, određeni minimalni broj njih koji je kvorum čvorova treba dogovoriti koji je jedan od poslužitelja *NameNode* stvarni primarni u bilo kojem trenutku. Ako se održava [7] neparan broj Zookeeper čvorova, ovaj bi kvorum mogao biti obična većina. Dakle, iako će to spriječiti nedosljednosti koje proizlaze iz scenarija podijeljenog mozga, mora postojati i rukovanje nadilaženjem. I tu se pojavljuje *Zookeeper FailoverController*. Ovo je jedna komponenta klijenta *Zookeeper* koja prati i upravlja stanjem *NameNode.* Nakon što je očito da određeni poslužitelj *NameNode* više nije funkcionalan, tada može započeti proces prelaska na pogrešku. Da bi to omogućili, primarni i *Standby NameNode* poslužitelji moraju održavati trajnu sesiju u *Zookeeperu*. Tako da čuvar *Zookeper* točno zna tko je od njih u zdravom stanju. I kako točno odlučuju je li čvor zdrav? Pa, ako čvor ne može održati sesiju, a ta sesija istječe. Tada će Zookeeper pretpostaviti da čvor ne može održavati sesiju zbog neke vrste problema, a zatim će se pokrenuti prebacivanje u stanje pripravnosti. Dakle, upravo ovo je način na koji *Zookeeper* omogućuje *NameNode* servisu u HDFS-u da radi u načinu visoke dostupnosti. Treba obratiti pažnju na još jednu uslugu u Hadoopu koja je predstavljala jednu točku kvara u starijim verzijama. Točnije, YARN ResourceManager. Prethodno je viđeno kako klijent može poslati MapReduce poslove YARN *ResourceManageru. ResourceManager* će tada biti svjestan svih resursa koji su dostupni u cijelom klasteru i tada će dodijeliti neke od tih resursa za svaki od poslova koje treba izvršiti.

Dijagram prikazuje da je klijentski proces povezan sa upraviteljem resursa strelicom. Ovo je zauzvrat povezano sa sva tri čvora strelicama (slika 3.11) Jasno je da je *ResourceManager* apsolutno ključan za rad *Hadoopa* i kao rezultat toga trebao bi biti vrlo dostupan. I baš kao i s uslugom *NameNode, Zookeeper* se također može koristiti kako bi *ResourceManager* bio vrlo dostupan. Dakle, rukovanje prelaskom na grešku za *ResourceManager* je slično načinu na koji *Zookeeper* upravlja prelaskom na grešku za *NameNode.* Dakle, kako bi osigurao da nema scenarija podijeljenog mozga, *Zookeeper* će koristiti zaključavanje kako bi osigurao da postoji samo jedan *ResourceManager* aktivan

u bilo kojem trenutku. Slično Zookeeper kontroleru za nadilaženje greške, koji se koristi s uslugom NameNode, postoji usluga ActiveStandbyElector. Koji je dio ResourceManagera i kontinuirano šalje status instancama Zookeepera.

Kao rezultat toga, ResourceManager također održava aktivnu sesiju s poslužiteljima Zookeeper. A ako bi sesija istekla, tada će ResourceManager u stanju pripravnosti biti promaknut u primarni. Dakle, s obzirom na to da su usluga NameNode i ResourceManager vrlo dostupni, Hadoop je puno otporniji na kvarove ovih ključnih komponenti. U bilo kojoj produkcijskoj postavci, te usluge ćete pokretati u načinu visoke dostupnosti, što je upravo razlog zašto Zookeeper nije pakiran zajedno s Hadoopom [7].



Slika 3.11 *Dijagram povezivanja klijentskog procesa sa upraviteljem resursa*

3.3.7 Kerberos protokol

Kerberos je protokol koji se primjenjuje kod provjere autentičnosti zahtjeva za uslugama između pouzdanih hostova preko nepouzdane mreže (npr. internet). Kerberos podrška je ugrađena u sve glavne računalne operativne sustave, uključujući *Windows, Microsoft, Apple macOS, FreeBSD i Linux.* Još od 2000 godine Microsoft koristi *Kerberos* protokol kao zadanu metodu provjere autentičnosti u Windows sustavu koji je sastavni dio usluge *Windows Active Directory(engl.AD).* Pružatelji širokopojasnih usluga također koriste protokol za provjeru autentičnosti kabelskih modema i set-top box uređaja koji pristupaju njihovim mrežama. Razvijen je u sklopu projekta *Athena* na *MIT-u (engl.Massachusetts Institue of Technologies).* Ime je preuzeto iz grčke mitologije Kerberos(Kerber). Tri glave Kerberos protokola predstavljaju sljedeće [8]:

1. Klijent
2. Mrežni resurs, koji je aplikacijski poslužitelj koji omogućuje pristup mrežnom resursu.
3. Centar za distirbuciju ključeva(KDC) koji djeluje kao Kerberosova pouzdana usluga provjere autentičnosti.

Slika 3.12 *Ilustracija Kerberos protokola*

3.3.7.1 Kako funkcionira Kerberos protokol za provjeru autentičnosti?

Slijedi pojednostavljeni opis rada protokola, budući da je stvarni proces kompliciraniji i može varirati od jedne implementacije do druge [8]:

1. Zahtjev poslužitelja za autentifikacijom. Kako bi se pokrenuo proces provjere autentičnosti *Kerberos* klijenta, inicirajući klijent šalje zahtjev za provjeru autentičnosti Kerberos KDC poslužitelju za provjeru autentičnosti. Početni zahtjev za autentifikaciju šalje se u obliku otvorenog teksta budući da u zahtjev nisu uključene osjetljive informacije. Poslužitelj za provjeru autentičnosti provjerava je li klijent u KDC bazi podataka i dohvaća privatni klijentov ključ.

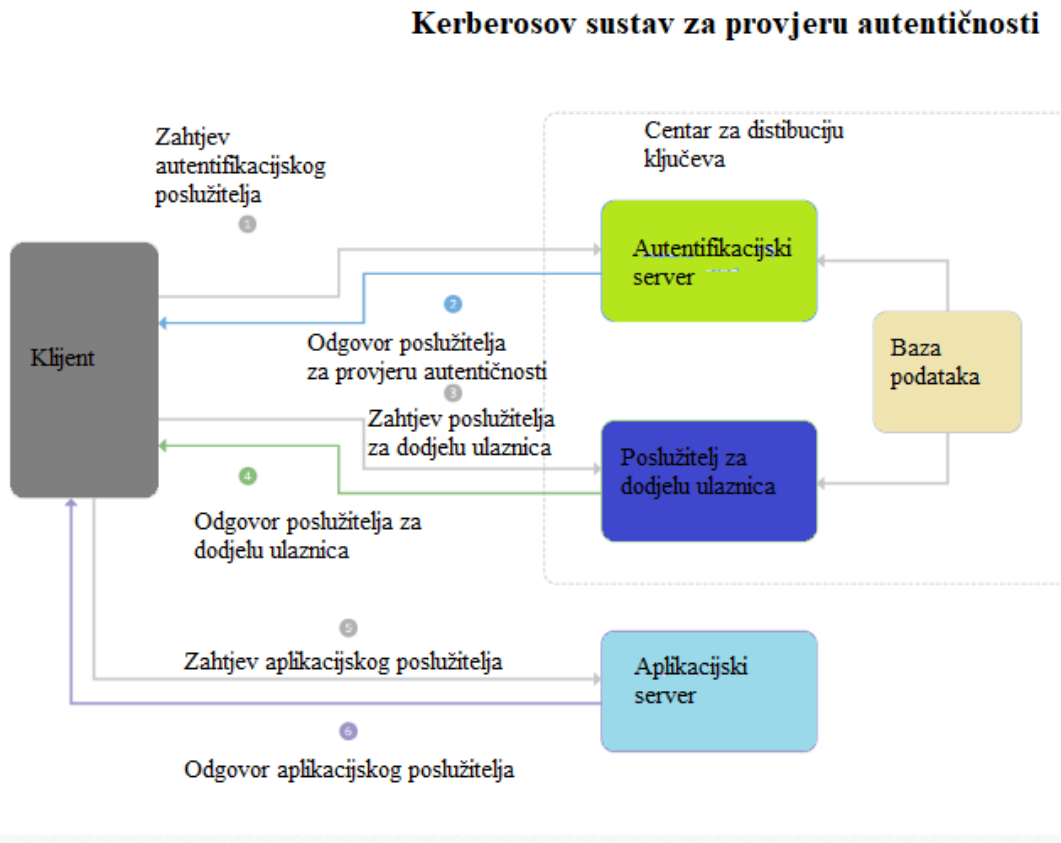
2. Odgovor poslužitelja za provjeru autentičnosti. Ako se korisničko ime inicijalnog klijenta ne pronađe u *KDC* bazi podataka, klijent se ne može autentificirati i proces autentifikacije se zaustavlja. Inače, autentifikacijski poslužitelj šalje klijentu *TGT* i ključ sesije.

3. Zahtjev za servisnu kartu. Nakon provjere autentičnosti od strane poslužitelja za provjeru autentičnosti, klijent traži servisnu kartu od *TGS-a*. Ovaj zahtjev mora biti popraćen *TGT-om* koji šalje *KDC* autentifikacijski poslužitelj.

4. Odgovor servisne karte. Ako TGS može autentificirati klijenta, šalje vjerodajnice i ulaznicu za pristup traženoj usluzi. Ovaj prijenos je šifriran ključem sesije specifičnim za korisnika i uslugu kojoj se pristupa. Ovaj dokaz identiteta koristi se za pristup traženoj "Kerberos" usluzi. Ta usluga provjerava izvorni zahtjev i zatim potvrđuje svoj identitet sustavu koji je zatražio.

5. Zahtjev aplikacijskog poslužitelja. Klijent šalje zahtjev za pristup aplikacijskom poslužitelju. Ovaj zahtjev uključuje servisnu kartu primljenu u koraku 4. Ako aplikacijski poslužitelj može autentificirati ovaj zahtjev, klijent može pristupiti poslužitelju.

6. Odgovor aplikacijskog poslužitelja. U slučajevima kada klijent zahtijeva od aplikacijskog poslužitelja da se autentificira, ovaj odgovor je potreban. Klijent se već autentificirao, a odgovor aplikacijskog poslužitelja uključuje *Kerberos* autentifikaciju poslužitelja



Slika 3.12 *Prikaz Kerberosov sustava za provjeru autentičnosti*

3.3.7.2 Konfiguracija Kerberos KDC i klijenta

Prije same konfiguracije potrebno je instalirati pakete kako bi konfiguracija bila omogućena. Koristeći naredbu *krb5-server* postavlja se zahtjev na poslužitelju te *krb5-workstation* na klijentu te na slici 3.12 slijedi prikaz instalacije [7]:

```
[root@s1 ~]# yum install krb5-server krb5-workstation pam_krb5
Loaded plugins: fastestmirror, langpacks
local                                                                    | 3.6 kB  00:00:00
Loading mirror speeds from cached hostfile
Resolving Dependencies
--> Running transaction check
---> Package krb5-server.x86_64 0:1.12.2-14.el7 will be installed
---> Package krb5-workstation.x86_64 0:1.12.2-14.el7 will be installed
---> Package pam_krb5.x86_64 0:2.4.8-4.el7 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================================================
 Package                  Arch              Version                    Repository          Size
================================================================================================================
Installing:
 krb5-server              x86_64            1.12.2-14.el7              local              905 k
 krb5-workstation         x86_64            1.12.2-14.el7              local              755 k
 pam_krb5                 x86_64            2.4.8-4.el7                local              158 k
```

Slika 3.13 *Instalacija paketa na poslužitelju i klijentu*

Nakon toga je potrebno pristupiti direktoriju *krb5dc* koristeći slijedeće naredbe na slici 3.14:

```
[root@s1 ~]#
[root@s1 ~]#
[root@s1 ~]# vi /var/kerberos/krb5kdc/
[root@s1 ~]# _
```

Slika 3.14 *Pristup direktoriju krb5dc*

Kada se u ovom direktoriju pokrene izlistavanje (*engl. ls command*) prikazat će se dvije datoteke *kadm5.acl* i *kdc.conf.* Ukoliko se pristupi *acl* datoteci može se primijetiti da admin uloga nije dodijeljena. S druge strane ukoliko se pristupi *kdc* datoteci može se primjetiti da se konfigurira port te se mora promijeniti domena (od *EXAMPLE* u *TUP*). Nakon što je to napravljeno, promjene je potrebno spremiti (slika 3.15).

```
[kdcdefaults]
 kdc_ports = 88
 kdc_tcp_ports = 88

[realms]
 TUP.COM = {
  #master_key_type = aes256-cts
  acl_file = /var/kerberos/krb5kdc/kadm5.acl
  dict_file = /usr/share/dict/words
  admin_keytab = /var/kerberos/krb5kdc/kadm5.keytab
  supported_enctypes = aes256-cts:normal aes128-cts:normal des3-hmac-sha1:normal arcfour-hmac:normal came
llia256-cts:normal camellia128-cts:normal des-hmac-sha1:normal des-cbc-md5:normal des-cbc-crc:normal
 }
~
~
~
```

Slika 3.15 *Konfiguracija kdc datoteke*

Slijedeći korak je konfiguracija u *etc* datoteci (slika 3.16).

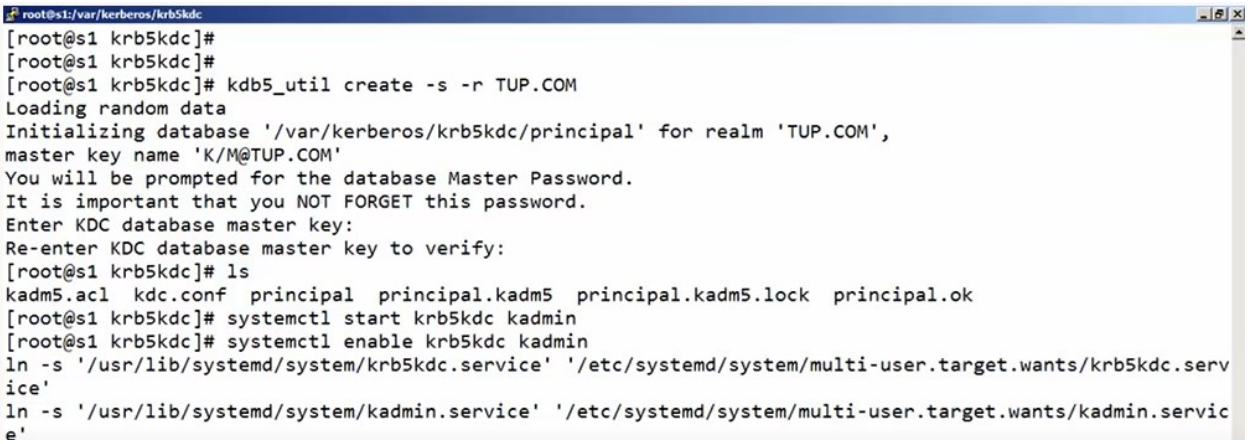
```
root@s1:/etc
[logging]
 default = FILE:/var/log/krb5libs.log
 kdc = FILE:/var/log/krb5kdc.log
 admin_server = FILE:/var/log/kadmind.log

[libdefaults]
 dns_lookup_realm = false
 ticket_lifetime = 24h
 renew_lifetime = 7d
 forwardable = true
 rdns = false
 default_realm = TUP.COM
 default_ccache_name = KEYRING:persistent:%{uid}

[realms]
 TUP.COM = {
 kdc = s1.tup.com
 admin_server = s1.tup.com
}

[domain_realm]
 .tup.com = TUP.COM
 tup.com = TUP.COM
~
~
```

Slika 3.16 *Konfiguracija etc datoteke*

Nakon toga se potrebno ponovno vratiti u prethodni direktorij koristeći naredbu cd- i potrebno je ponovno izvršiti izlistavanje (slika 3.17).

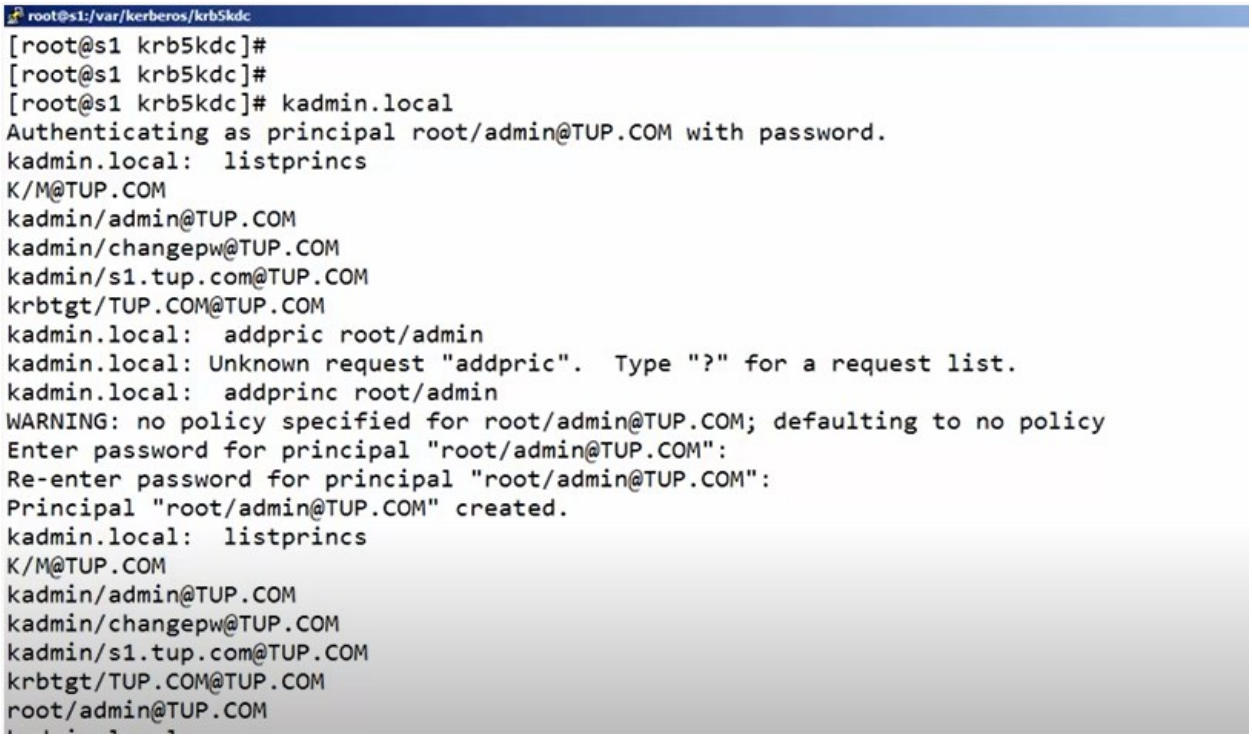
Slika 3.17 *Prikaz vraćanja u prethodni direktorij*

Nakon toga je potrebno inicijalizirati i kreirati bazu podataka te je u isto vrijeme potrebno pokrenuti generator slučajnih brojeva, koji već mora biti instaliran ranije. Ukoliko to nije napravljeno pojavit će se poruka o pogreški. Slijedeći korak koji je potrebno napraviti je unijeti KDC master ključ. Kada se ponovno pokrene izlistavanje mogu se vidjeti novokreirane baze podataka. Koristeći naredbu *systemctls start krb5kdc kadmin* pokreće se krb5dc direktorij te se slijedećom naredbom enable omogućuje usluga.



Slika 3.18 *Inicijaliziranje i kreiranje baze podataka*

Nakon inicijalizacije baze podataka potrebno je dodavati određene unose u Keberos bazu podataka. Kerberos dozvoljava autentifikaciju. Nakon autentifikacije započinje raščlanjivanje tokena. Koristi se naredba *kadmin.local* za autentifikaciju. Potrebno je unijeti neke dodatne unose koristeći naredbu *addprinc root/admin* . Također je potrebno postaviti lozinku za kdc bazu podataka.

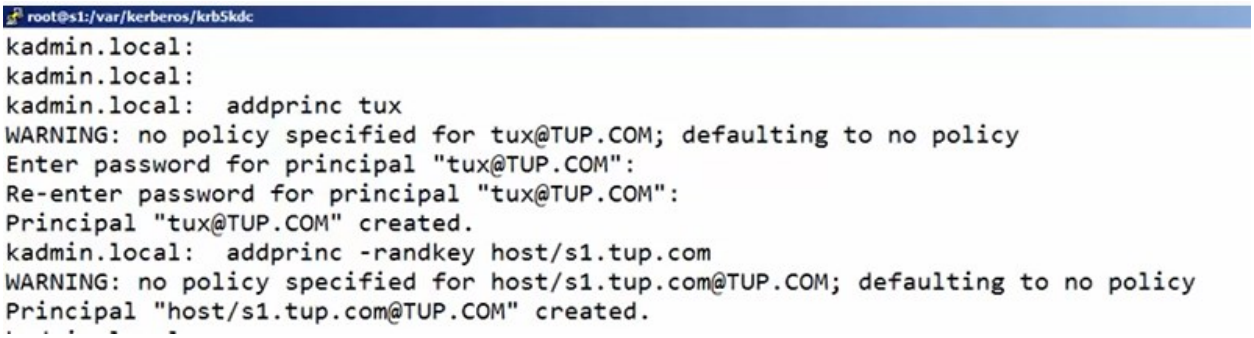


Slika 3.19 *Dodavanje unosa u bazu podataka*

Kako bi se omogućio pristup korisničkom računu koji je kreiran prethodno, neovisno o tome koji je njegov naziv mora se kreirati okvir za taj račun u *Kerberos* bazi podataka (u ovom primjeru je kreiran korisnički račun tux).

Slika 3.20 *Dodavanje korisničkog računa u Kerberos bazu podataka*

Nakon što je dodan korisnički račun potrebno je unijeti lozinku. Ideja je da se nakon toga može ulogirati i izabrati odgovarajući *Kerberos* token i prezentirati *Kerberos* token koji je siguran za *kdc* datoteku. Potrebno je postaviti načela za navedeni server i dodati slučajni ključ(slika 3.21).



Slika 3.21 *Postavljenje načela za server i dodavanje slučajnog ključa*

Nakon postavljanja načela, naredbom listprincs se mogu izlistati(slika 3.22).

Slika 3.22 *Izlistavanje postavljenih načela na serveru*

Potrebno je kopirati *Kerberos* bazu podataka u etc direktorij koristeći naredbu *ktadd host/s1.tup.com_(slika 3.23).*

```
kadmin.local:  ktadd host/s1.tup.com
Entry for principal host/s1.tup.com with kvno 2, encryption type aes256-cts-hmac-sha1-96 added to keytab
FILE:/etc/krb5.keytab.
Entry for principal host/s1.tup.com with kvno 2, encryption type aes128-cts-hmac-sha1-96 added to keytab
FILE:/etc/krb5.keytab.
Entry for principal host/s1.tup.com with kvno 2, encryption type des3-cbc-sha1 added to keytab FILE:/etc/
krb5.keytab.
Entry for principal host/s1.tup.com with kvno 2, encryption type arcfour-hmac added to keytab FILE:/etc/k
rb5.keytab.
```

Slika 3.23 *Kopiranje Kerberos baze podataka u etc direktorij*

Nakon kopiranje *Keberos* baze u *etc* direktorij potrebno je konfigurirati Kerberos autentifikaciju preko klijenta (popis naredbi je naveden na slici 3.24).

```
root@s1:/var/kerberos/krb5kdc
[root@s1 krb5kdc]#
[root@s1 krb5kdc]#
[root@s1 krb5kdc]# vi /etc/ssh/ssh_
```

Slika 3.24 *Konfiguracija Kerberos autentifikacije preko klijenta*

Ispis nakon konfiguracije se može vidjeti na slici 3.25.

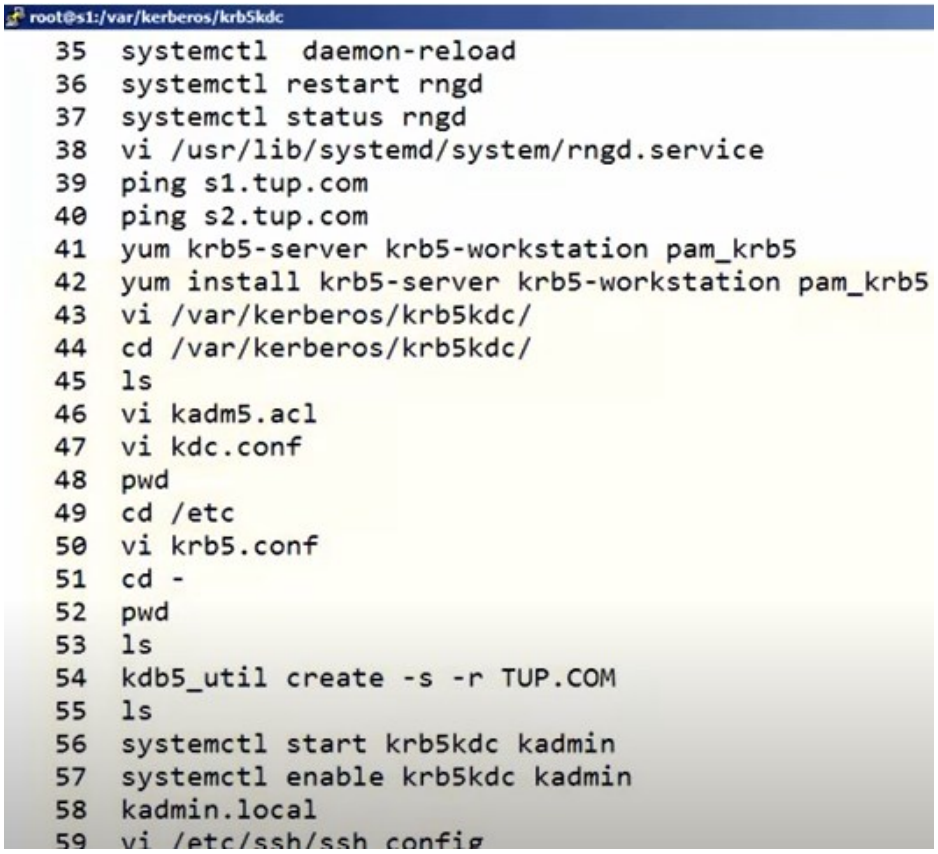
```
root@s1:/var/kerberos/krb5kdc
#       $OpenBSD: ssh_config,v 1.28 2013/09/16 11:35:43 sthen Exp $

# This is the ssh client system-wide configuration file.  See
# ssh_config(5) for more information.  This file provides defaults for
# users, and the values can be changed in per-user configuration files
# or on the command line.

# Configuration data is parsed as follows:
#  1. command line options
#  2. user-specific file
#  3. system-wide file
# Any configuration value is only changed the first time it is set.
# Thus, host-specific definitions should be at the beginning of the
# configuration file, and defaults at the end.

# Site-wide defaults for some commonly used options.  For a comprehensive
# list of available options, their meanings and defaults, please see the
# ssh_config(5) man page.

# Host *
#   ForwardAgent no
#   ForwardX11 no
#   RhostsRSAAuthentication no
#   RSAAuthentication yes
```

Slika 3.25 *Ispis nakon konfiguracije*

Kako bi se postavio klijent na serveru i omogućila njegova autentifikacija koristi se naredba *authconfig—enablekrb5—update.* Na taj se način omogućuje *ssh* serveru prihvaćanje *Kerberos* autentifikacije pri čemu je *ssh* server svijestan svih mogućih promjena.

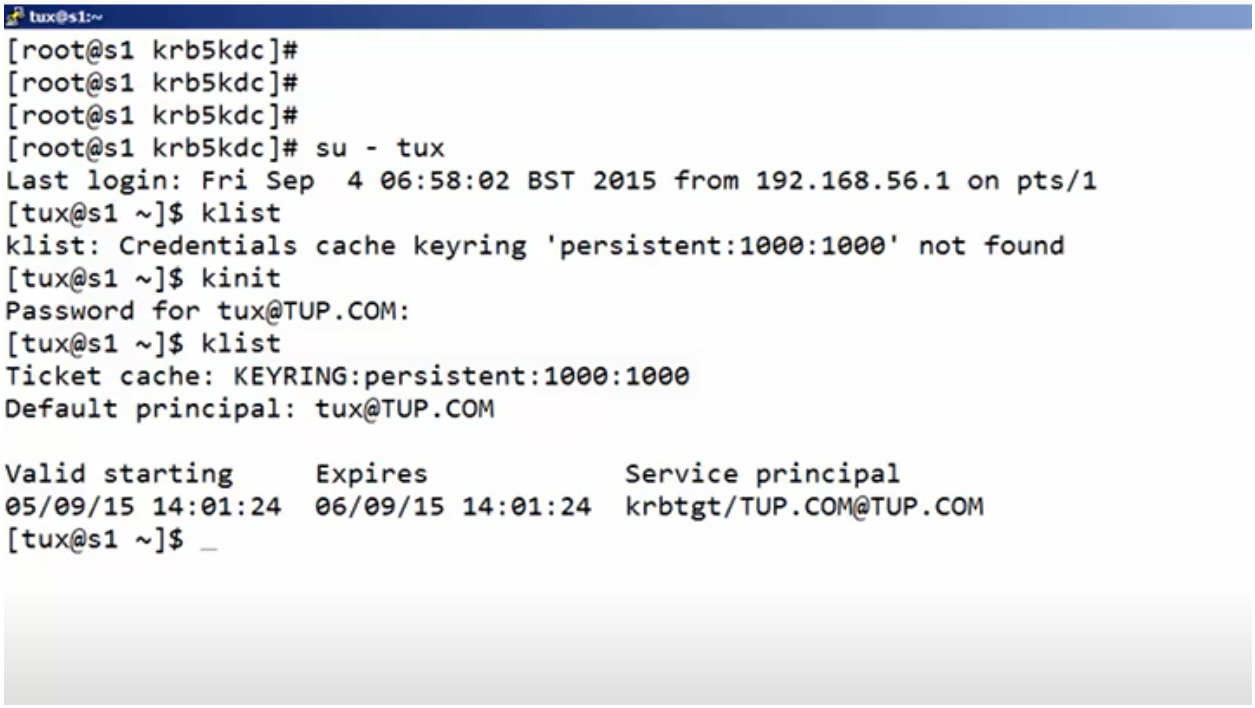
Slika 3.26 *Omogućavanje ssh serveru prihvaćanje Kerberos autentifikacije*

Kada se upiše naredba history u retku 42 se može vidjeti da se postavlja *Kerberos* server, također se instaliraju elementi za radnu stanicu. U retku 44 prolazi se kroz var Kerberos strukturu. Prolazeći kroz izlistavanje mogu se vidjeti konfiguracijski elementi. Sve reference su promijenjene iz EXAMPLE u TUP. Svi prethodno napravljeni koraci mogu se vidjeti na slici 3.27.



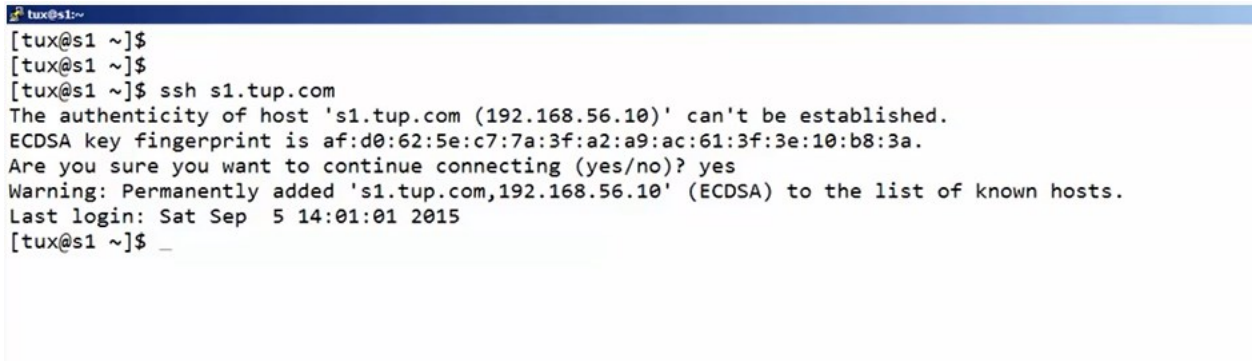
Slika 3.27 *Prikaz svih prethodno napravljenih koraka u konfiguraciji KDC i klijenta*

Ukoliko se prebaci na običan korisnički račun, u ovom primjeru je kreiran račun imena *tux*. Koristi se naredba *kinit* kako bi se povezao na *Kerberos* server i kako bi se dohvatio token. Nakon toga se unosi lozinka za *Kerberos*. Kada se upiše naredba *klist* može se vidjeti dohvaćeni token, token se čuva 24h (slika 3.28).



Slika 3.28 *Povezivanje na Kerberos server i dohvačanje tokena*

Što znači da jednom kad se korisnik autentificira na *Kerberos* server prenosi token i ostale resurse na njega (slika 3.29).



Slika 3.29 *Prikaz prenošenja tokena i ostalih resursa na Kerberos server*

4. UPRAVLJANJE DIGITALNIM IDENTITETIMA U OBLAKU

U drugom poglavlju opisane su tehnologije za upravljanje identitetima u oblaku, dok je u trećem poglavlju raspravljano o sigurnim tehnologijama računarstva u oblaku. Sve više dolazi do izražaja činjenica kako je za siguran rad oblaka ključno učinkovito upravljanje identitetom. Međutim, veliki dio nedavnih istraživanja je usmjeren na jednostavne modele kontrole pristupa temeljene na modelima za sigurnost u oblaku [9]. U okruženju u kojem se oblak nalazi će biti teško definirati korisničke uloge u organizaciji. Sam korisnik može imati nekoliko identiteta preko više sustava, ali i preko više oblaka. Prema *Gopalakrishnanu* [9] upravljanje identitetom u oblaku se mora temeljiti na upravljanju kontrolnim točkama u dinamičkom kompozitu povučenih strojeva, virtualnih uređaja ili usluga identiteta. Usluge u oblaku mogu biti dinamične u prirodi, stoga pri upravljanju životnim ciklusima identiteta treba uzeti u obzir aspekte kao što su pružanje usluge i onemogućavanje usluge, standardi kao što je Service Markup Language, jezik koji se koristi za web usluge mora uzeti u obzir značajke oblaka kao što je dodjela resursa u stvarnom vremenu. Korisnikovi id-evi u oblaku će biti dinamični te će i tehnologije kao što je primjerice OpenID biti usmjerene na funkcionalnosti u oblaku. Prilikom prilagođavanja OpenID-a najveći problem koji se javlja je problem uspostavljanja odnosa u oblaku, stoga je odgovarajući model povjerenja ključan. Kao što je ranije već navedeno, jednostavna kontrola pristupa temeljena na ulogama (engl.RBAC) previše je ograničavajuća za okruženje u oblaku. Kontrola pristupa temeljena na atributima(ABAC), na kojima se temelje pojedini standardi kao što je primjerice XACML koji je široko rasprostranjen servis za orijentirane sustave. Prema Oldenu [8] RBAC i ABAC nisu prikladni za oblak. To je zato što u okruženju oblaka, atributi i uloge članstva su odvojeni od poslovanja sustava i mogu se distribuirati kroz sustave putem federacije. Kada se govori o autentifikaciji, jedinstvene prijave s Assertions Markup Language se ističu u okruženju usmjerenom na usluge. Međutim, cijena izgradnje infrastrukture identiteta temeljena na SAML-u je opsežna i treba se ispitati njegova uporaba u oblaku. Ostale usluge koje digitalni identiteti moraju osigurati su revizija i odgovornost. Revizija u distribuiranom okruženju dolazi s brojnim izazovima, postavlja se pitanje koliko revizijskih podatka prikupljati i koje tehnike koristiti za analizu podataka. U okruženju oblaka, revizija će postati izazovnija, posebice u javnom oblaku. Jedan od ciljeva oblaka je dinamička raspodjela resursa bez obzira na to tko je tražio resurse i gdje ti resursi mogu biti. U takvom okruženju hvatanje svih aktivnosti oblaka postaje izazov. Autori [9] predlažu semantičke

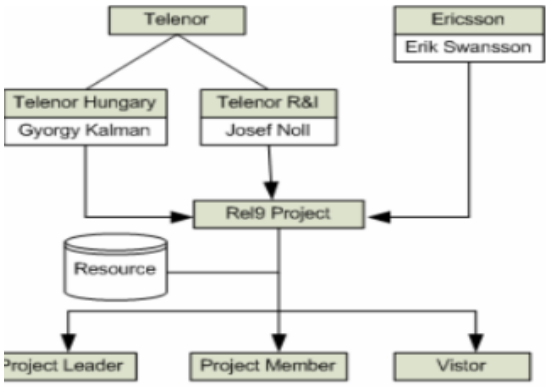
tehnologije kako bi se navedeni problemi riješili. Kako bi se naveden usluge mogle pružiti, formulirana su pravila i politika za kontrolu pristupa resursima u vlasništvu grupe ili zajednice. Navedene informacije se moraju kodirati kako bi se olakšalo manipulacija i razumijevanje računalnih programa. Ovo kodiranje se provodi pomoću specifikacije i korištenja ontologije, formalnog predstavljanja domena. Koristeći semantičke tehnologije, predlaže se korištenje SemID(semantičkog identiteta) ontologije za rješavanje rastuće potrebe za upravljanje identitetom i podrškom za privatnost u korporativnim mrežama. Tehnologija semantičkog weba je slijedeća generacija suvremenog weba. Veliki broj firmi poput Adobe-a, Googl-a, HP i sl se temelje na semantičkom webu [10]. *Chowdury*[10] u svom radu ističe kako se upravljanje digitalnim identitetima sastoji od osobnog identiteta (*PID (engl.personal identifiers))* i sigurnosnog identiteta *(SID(enlg.security identifier)).*

PID-ovi se koriste za identifikaciju u osobnim i komercijalnim interakcijama. Slično tomu, SID-ovi i CID-ovi se mogu koristiti u osobnim i profesionalnim interakcijama. SemID se fokusira na upravljanje digitalnim identitetom u korporativnom okruženju. Današnji ljudi sve više rade u projektno orijentiranim grupama. Takvu grupu mogu formirati članovi različitih partnerskih organizacija. *OWL Webtechnology Language* se koristi za formaliziranje i definiranje predložene domene upravljanja digitalnim identitetom. Namjera je učiniti informacije sadržane u ovoj domeni spremnima za obradu u budućim aplikacijama. *OWL* je odabran jer omogućuje veću strojnu interpretabilnost web sadržaja od one koju podržavaju XML, RDF i RDF shema (RDF-S) pružajući dodatni vokabular zajedno s formalnom semantikom. Među podjezicima OWL-a korišten je *OWL DL* zbog njegove računalne potpunosti i mogućnosti odlučivanja [9].

4.1 Slučaj upotrebe digitalnih identiteta: upravljanje korporativnim identitetom

Slika 4.1 prikazuje primjer slučaja uporabe digitalnog identiteta. U *Telenor AS* kreiran je novi projekt pod nazivom *Rel9 (Release 9) Project*. Čine ga *Gyorgy Kalman, Josef Noll* i *Erik Swansson* kao članovi. Predstavljaju *Telenor Mađarska,* Telenor R&I i Ericsson u projektu Rel9. Dakle, oni su dio svoje grupe/tvrtke i novonastalog projekta. Projekt ima neke resurse kao što su web stranica, podaci o članu, proračun, razni dokumenti i rezultati itd. Jedan od članova služit će kao voditelj projekta, a ostali članovi će raditi kao obični članovi. Posjetitelji su sve osobe iz *Telenora* ili *Ericssona* koje nisu članovi *Rel9* projekta, a žele se informirati o projektu i njegovom statusu. Na

temelju opisa slučaja uporabe i zahtjeva, formulirat će se *SemID* ontologija i instance [10].



Slika 4.1 *Primjer scenarija uporabe digitalnog identiteta*

4.2 Upravljanje digitalnim identitetom koristeći semantiku

Prihvaćen je prijedlog da se *SemID* bavi upravljanjem identitetom ljudi u profesionalnom životu. Također se očekuje da će osigurati kontrolu pristupa i usluge privatnosti na temelju uloga koje imaju u radnom okruženju temeljenom na projektu. Kontrola pristupa i ciljevi privatnosti postižu se politikama i pravilima. U ovom radu će biti opisan *OWL DL* koji se koristi za formalizaciju semantike predloženog upravljanja identitetom [10].

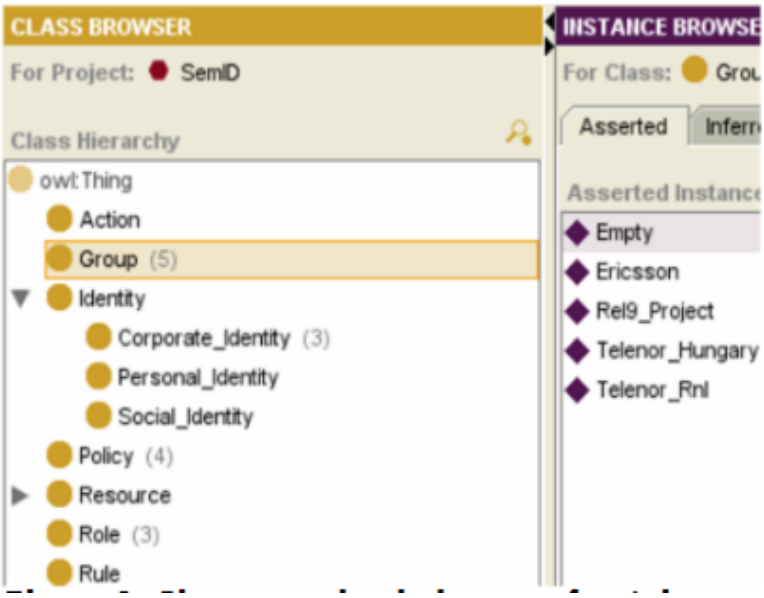
4.3 Funkcionalna arhitektura

U ontologiji je jasno definiran korporativni identitet svakog člana i projektne skupine kojoj pripada. Svakom članu projekta je dodijeljena određena uloga poput voditelja projekta, člana projekta i posjetitelja. Svaka uloga ima određenu politiku (ili politike). Politika predstavlja privilegiju rezerviranu za svaku ulogu u zajednici i izraženu kroz skup pravila. Pravilo je u biti funkcija koja uzima zahtjev za pristup kao ulaz i rezultira radnjom (dopusti, odbij ili ne-aplikacija). Da bi se utvrdilo je li pravilo primjenjivo na

zahtjev za pristup, koristi se cilj koji se sastoji od nekoliko elemenata. Cilj je skup pojednostavljenih uvjeta za predmet, resurs i radnju koji moraju biti ispunjeni da bi se pravilo primijenilo na dati zahtjev [10]. U predloženoj SemID ontologiji, subjekti su identiteti (CID-ovi) koji igraju specifične uloge poput voditelja projekta, člana projekta i posjetitelja. Resursi su resursi projekta. Teoretski, pravila sadrže trostruki cilj, radnju i politiku. Stoga je pravilo pojednostavljeno kao trostruki predmet, resurs i radnja. Obični član Rel9 projekta ima samo dopuštenje za čitanje i pisanje. Posjetitelju treba zabraniti pisanje preko projektne dokumentacije. Na ovaj način se problemi s kontrolom pristupa rješavaju u SemID-u. Zahtjevi privatnosti zadovoljeni su u SemID ontologiji pomoću dva svojstva (hasVisibility i hasVisibilityOfGroup). hasVisibilityOfGroup je pripojen klasi: Role, a hasVisibility je pripojen klasi: Identity (CID). Posljednje svojstvo je općenito svojstvo vidljivosti koje osigurava da svatko (tko ima CID) pripada nekim grupama ima vidljivost resursa tih grupa. Dok vidljivost temeljena na ulogama predstavlja specifičniju vidljivost resursa. Na temelju uloga; voditelj, članovi ili posjetitelji projekta imaju različitu vidljivost detalja o članu. Voditelj i članovi Rel9 projekta imaju vidljivost detalja sučlana koje posjetitelji ne mogu vidjeti. hasVisibilityOfGroup to osigurava u SemID-u. Stoga se kontrola pristupa resursima projekta (isključujući podatke o članu) održava putem politika i pravila [10].

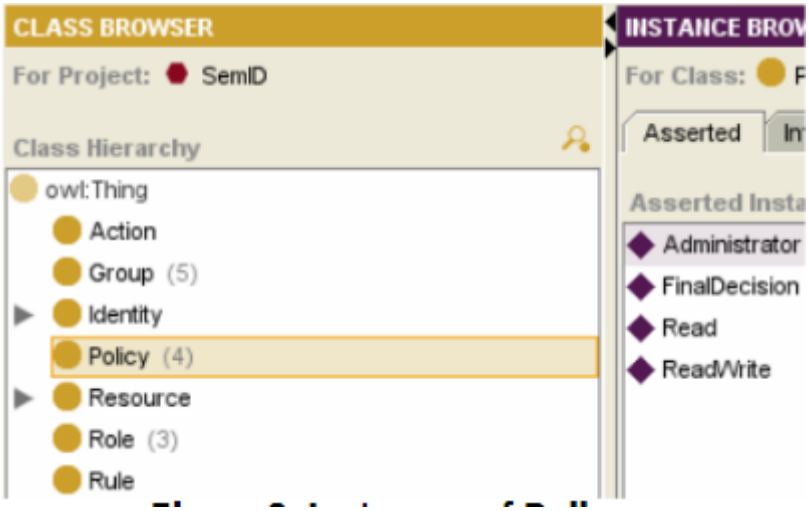
4.4 Implementacija

Modelira se ontologija prema scenariju korištenja s *OW-DL* temeljen na *Protege* ontološkoj editor platformi. Na slici 4.2 s lijeve strane se može vidjeti prikaz klasa i podklasa *SemID* ontologije za modeliranje predloženog slučaja uporabe i ispunjavanje zahtjeva [9].



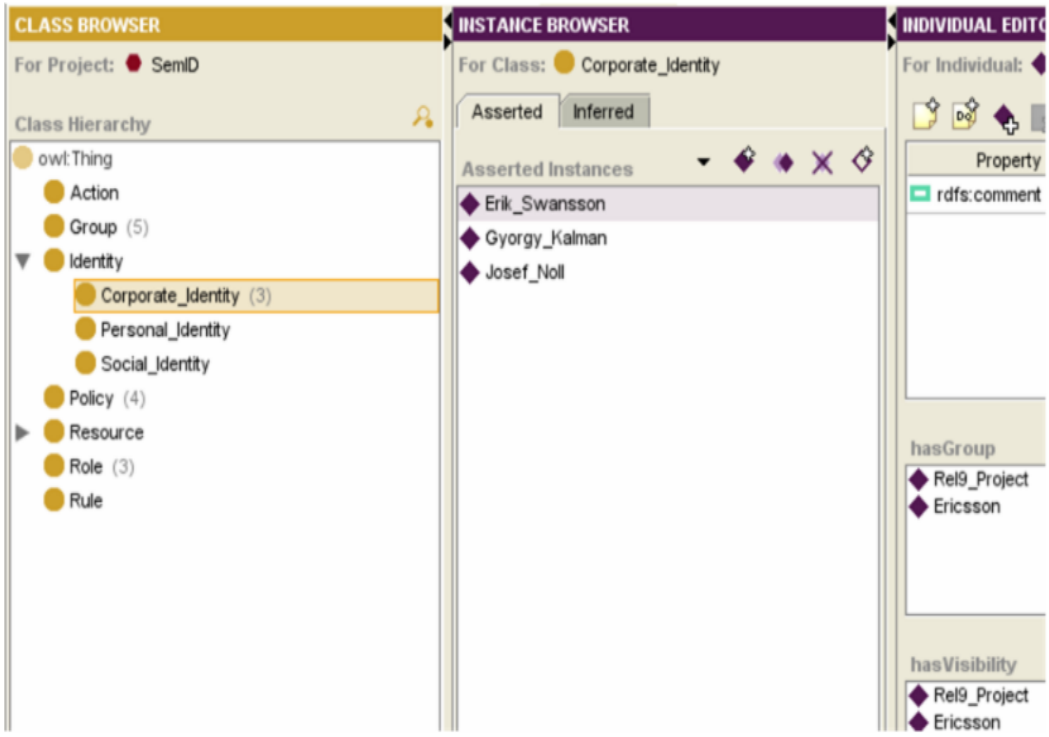
Slika 4.2 *Klase i podklase ontologije*

Slika 4.2 također ilustrira instance četiriju različitih grupa opisanih u scenariju slučaja upotrebe. Prazna grupa je stvorena da podrži privatnost grupe kada posjetitelji pokušaju pristupiti resursima. Slika 4.3 ilustrira četiri različite politike uloga: Administrator, Konačna odluka, Čitanje i Čitanje i pisanje.



Slika 4.3 *Primjer korištenja različith politika*

U prikazanoj ontologiji, korporativni identitet (*engl.CID*) svakog predstavnika iz *Telenor R&I, Telenor Mađarska* i *Ericsson* definiran je njegovim imenom. Oni predstavljaju instance *CID* podklase. Svaka instanca ima dva svojstva hasGroup i hasVisibility. Grupa kojoj član pripada eksplicitno se identificira pomoću svojstva hasGroup. hasVisibility ukazuje na grupe čiji član zahtijeva vidljivost. Na primjer, osigurava da 'Erik Swansson ima vidljivost *Ericssona* i *Rel9* projektne skupine budući da je član obje ove skupine. Slika 4.4 daje primjer ovih definicija [9].



Slika 4.4 *Instance korporativnog identiteta i njihova svojstva.*

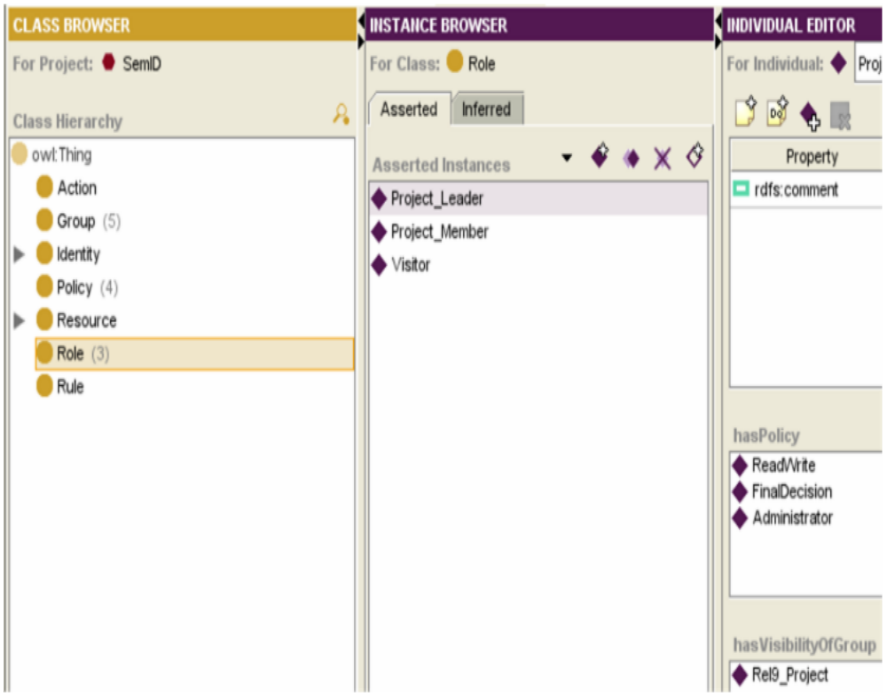
Postoje tri moguće uloge u *Rel9* projektu. Slika 4.5 prikazuje instance klase *Role.* Svaka uloga ima specifičnu politiku koja je izražena svojstvom *hasPolicy*. Odgovarajuće politike dodaju se svakoj instanci *Role*. U skladu s ciljem opisanim u prethodnom odjeljku, voditelj projekta može imati politike kao što su administrator, konačna odluka i *ReadWrite*. Kako bi se zadovoljili zahtjevi privatnosti grupe, stvoreno je svojstvo *hasVisibilityOfGroup.* Instanca posjetitelja ima vidljivost prazne grupe (instanca Grupe). Osigurava da bi korisniku kao posjetitelju trebalo biti dopušteno čitati dokumente projekta, ali nema dopuštenje vidjeti pojedinosti o članovima posjećenog

projekta. Ali kao članu projekta dopušteno mu je vidjeti podatke o svojim sučlanovima. Tablica 1 daje popis svojstava koja se koriste u ovoj ontologiji. Svako svojstvo ima svoju domenu i raspon. Klase kojima je pridruženo svojstvo nazivaju se domena. Dopuštene klase za svojstva često se nazivaju rasponom svojstva. Ova svojstva povezuju klase kroz sljedeće činjenice:

- Identitet ima grupu (*hasGroup).*
- Identitet ima vidljivost (*hasVisibility).*
- Grupa ima ulogu *(hasRole).*
- Uloga ima politiku (*hasPolicy).*
- Uloga ima vidljivost grupe (*hasVisibilityOfGroup).*
- Politika ima pravilo (*hasRule).*
- Pravilo ima predmet (*hasSubject).*
- Pravilo ima resurs (*hasResource).*
- Pravilo ima radnju *(hasAction), hasVisibility i hasVisibilityOfGroup* osiguravaju privatnost grupa. *hasSubject, hasResource i has Action* stvaraju pojednostavljeno pravilo

Tablica 2. *Popis svojstava, njihovih domena i raspona*

Svojstva	**Domena**	**Raspon**
hasGroup	identitet	grupa
hasVisibility	identitet	grupa
hasRole	grupa	uloga
hasPolicy	uloge	politika
hasVisibilityofGroup	uloge	grupa
hasSubject	pravila	pravila
hasResource	pravila	izvor
hasAction	pravila	akcija



Slika 4.5 *Instance uloga i njihovih svojstava*

5. STANDARDI ZA UPRAVLJANJE IDENTITETIMA U OBLAKU I NEDOSTATCI

Dok su *W3C* i *OASIS* razvili nekoliko standarda za upravljanje digitalnim identitetom, web usluge, *XML,XACML,SAML* i semantički web, standardi za sigurne promjene su tek u početcima svog razvoja. Međutim, OASIS je nedavno formirao tehnički odbor(TC)za identitet upravljanja privatnosti i povjerenja u računarstvu u oblaku. Kao što je navedeno u [10], *OASIS IDCloud TC* radi na rješavanju ozbiljnih sigurnosnih izazova postavlja upravljanje identitetom u računalstvu u oblaku.TC identificira nedostatke u postojećem identitetu standarde upravljanja i istražuje potrebu za profila za postizanje interoperabilnosti unutar trenutne standardima. Izvodi analizu rizika i prijetnji ,prikuplja slučajeve uporabe i daje smjernice za ublažavanje ranjivosti. TC je izjavio da će koristiti različite OASIS standarde kao njihove sastavne dijelove za razvijanje standarda upravljanja identitetom u oblaku. Ti građevni blokovi uključuju sljedeće usluge digitalnog

potpisa: proširivi resursIdentifikator (XRI) i XRI razmjena podataka. Kategorije slučajeva upotrebe koje je ispitao ovaj TC uključiti uspostavljanje povjerenja infrastrukture : upravljanje identitetom infrastrukture, udruženo identificirati upravljanje, autentifikaciju, autorizaciju, upravljanje računom/atributima, sigurnosni tokeni, revizija i sukladnost.

5.1 Upravljanje identitetom u oblaku: sigurnosni izazovi

Upravljanje identitetom široka je domena koja uključuje mnoga otvorena sigurnosna pitanja i izazove uključujući otvorenost, krađu identiteta, najmanje privilegije, povišenu kontrolu privilegija, dostupnost, povjerljivost, integritet, upravljanje povjerenjem itd [12]. Budući da se tehnološko područje kao što je *Cloud Identity Management* još uvijek razvija, stoga do sada nisu razvijeni dobro uspostavljeni standardi, što je rezultiralo problemima poput vezanosti dobavljača i nedostatka otvorenosti (dostupnost specifikacija/formata podataka/protokola). Otvorenost je ključna, tako da se pretvarači mogu razviti u budućnosti kako bi se osigurala interoperabilnost i skalabilnost. Nadalje, budući da podacima o identitetu mogu pristupiti neovlašteni ili zlonamjerni korisnici/uljezi. Stoga su potrebni odgovarajući sigurnosni (enkripcija, kontrola pristupa autentifikaciji, nadzor itd.) mehanizmi za pristup identitetu i administrativnim sučeljima; inače bi neovlašteni pristup mreži (prisluškivanje) bio ključni faktor rizika [12].Uz to, *Cloud IDMS* se mora pretplatiti na načelo najmanje privilegije i nastaviti korištenjem mehanizma tijeka rada za dodatna odobrenja. Međutim, budući da najmanje privilegije ne uključuju samo statične uloge i resurse, već i složen kontekst i dinamičke promjene (i tehničke i ne-tehničke). Stoga mnoge današnje implementacije *Cloud IDMS-a* enormno pretjerano osiguravaju učinkovita prava pristupa korisnicima kojima ta prava pristupa čak i nisu potrebna. Takvo pretjerano osiguravanje pristupa dovodi do brojnih kritičnih sigurnosnih problema uključujući neovlašteno otkrivanje, prijevaru, slučajni pristup i krađu identiteta.

Drugi najkritičniji aspekt upravljanja identitetom je upravljanje pristupom povlasticama. Osim tradicionalnih zahtjeva revizije i zapisivanja, pristup povlasticama također treba biti mehanizam upravljanja. Budući da većina poslovnih aplikacija pati od lošeg upravljanja ulogama, kršenja podjele dužnosti, izvješćivanje (i o uspješnim i

neuspješnim događajima) o ovom aspektu postaje ključno. Štoviše, napadi na usluge identiteta ili mrežnu povezanost, kao što su DDoS napadi ili iscrpljivanje resursa, mogu ugroziti dostupnost ili pogoršati performanse IDMS-a. Ako je potrebna visoka dostupnost i/ili izvedba, moraju se razmotriti opcije redundancije i failover. Osim toga, zbog sveprisutne prirode Clouda, njemu se može pristupiti putem raznih uređaja i aplikacija što rezultira povećanim brojem pristupnih točaka, što prije ili kasnije povećava prijetnju neovlaštenog otkrivanja (povjerljivosti) vjerodajnica identiteta od oba insajdera (osoblje) i autsajderi (korisnici/uljezi)]) [12]. Slično tome, s obzirom na povećani broj pristupnih točaka i entiteta sustava, cjelovitost podataka o identitetu i informacija pohranjenih u oblaku još je jedan važan problem na koji treba odmah obratiti pozornost [12].Uz gore spomenuta pitanja, upravljanje povjerenjem između pružatelja Cloud identiteta i pretplatnika jedno je od glavnih pitanja s kojima se današnji Cloud IDMS-ovi suočavaju. Nadalje, povjerenje je subjektivan i kontekstualno osjetljiv pojam koji još više otežava odabir pružatelja Cloud identiteta s potpuno pouzdanim uslugama identiteta [12].

Trenutno je veliki broj istraživanja usmjeren na sustave upravljanja identitetom u oblaku; međutim, sigurnost *Cloud IDMS-ova* je aspekt koji je još uvijek u početnim fazama i zahtijeva daljnje istraživanje. Postojeći IDMS-ovi podložni su različitim sigurnosnim i uskim grlima u izvedbi, što ograničava njihovu široku primjenu kao potencijalnog rješenja za dinamičko okruženje Clouda. Predstavljen je opis napada koji su ili pokrenuti protiv IDMS-ova ili koriste identitet kao glavni alat za napad. Kratak opis svih identificiranih napada predstavljen je u tablici 1. Osim toga, dodijeljena je jedinstvena oznaku svakom identificiranom napadu koji se kasnije koristi za označavanje navedenih napada. Sastavljeni popis pomogao je u određivanju ključnih sigurnosnih značajki koje IDMS temeljen na oblaku mora pružiti kako bi se osigurala zaštita i sigurnost vjerodajnica identiteta u oblaku.

Uz napade, također su identificirane značajke za Cloud IDMS-ove koje mogu ublažiti utjecaj većine ako ne i svih ovdje spomenutih napada. Uz popis eminentnih značajki Cloud IDMS-a, također je osiguran popis mehanizama koji pomažu u postizanju identificiranih značajki. Snage i slabosti svakog mehanizma također su specificirane gdje god je potrebno. Značajke Cloud IDMS-a zajedno s mehanizmima za implementaciju ovih značajki predstavljene su u obliku dobro organizirane taksonomije [12].

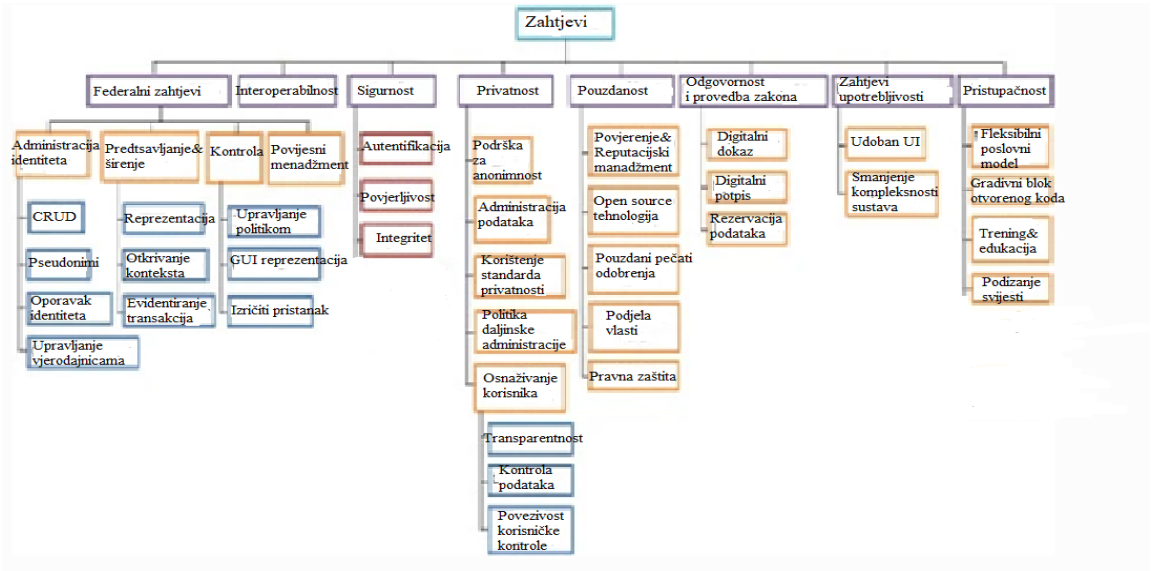
Tablica 3. *Poznati napadi na Cloud IDMS-ove*

Oznaka	Napad	Opis
A1	Napad grubom silom	Metoda hakiranja koja koristi pokušaje i pogreške za probijanje lozinki, vjerodajnica za prijavu i ključeva za šifriranje.
A2	Napad ponavljanja kolačića	Događa se kada napadač ukrade važeći kolačić korisnika i ponovno ga upotrijebi kako bi oponašao tog korisnika za izvođenje lažnih ili neovlaštenih transakcija/aktivnosti.
A3	Napad petljanja podataka	Neovlaštenim mijenjanjem podataka namjerno se mijenjaju i manipuliraju podatci putem ovlaštenih kanala.
A4	Napad uskraćivanjem usluge(DoS)	(DoS) napad je napad namijenjen gašenju stroja ili mreže, čineći ga nedostupnim korisnicima kojima je namijenjen. DoS napadi to postižu preplavljivanjem mete prometom ili slanjem informacija koje pokreću rušenje.
A5	Prisluškivanje	Napad prisluškivanjem događa se kada haker presretne, izbriše ili

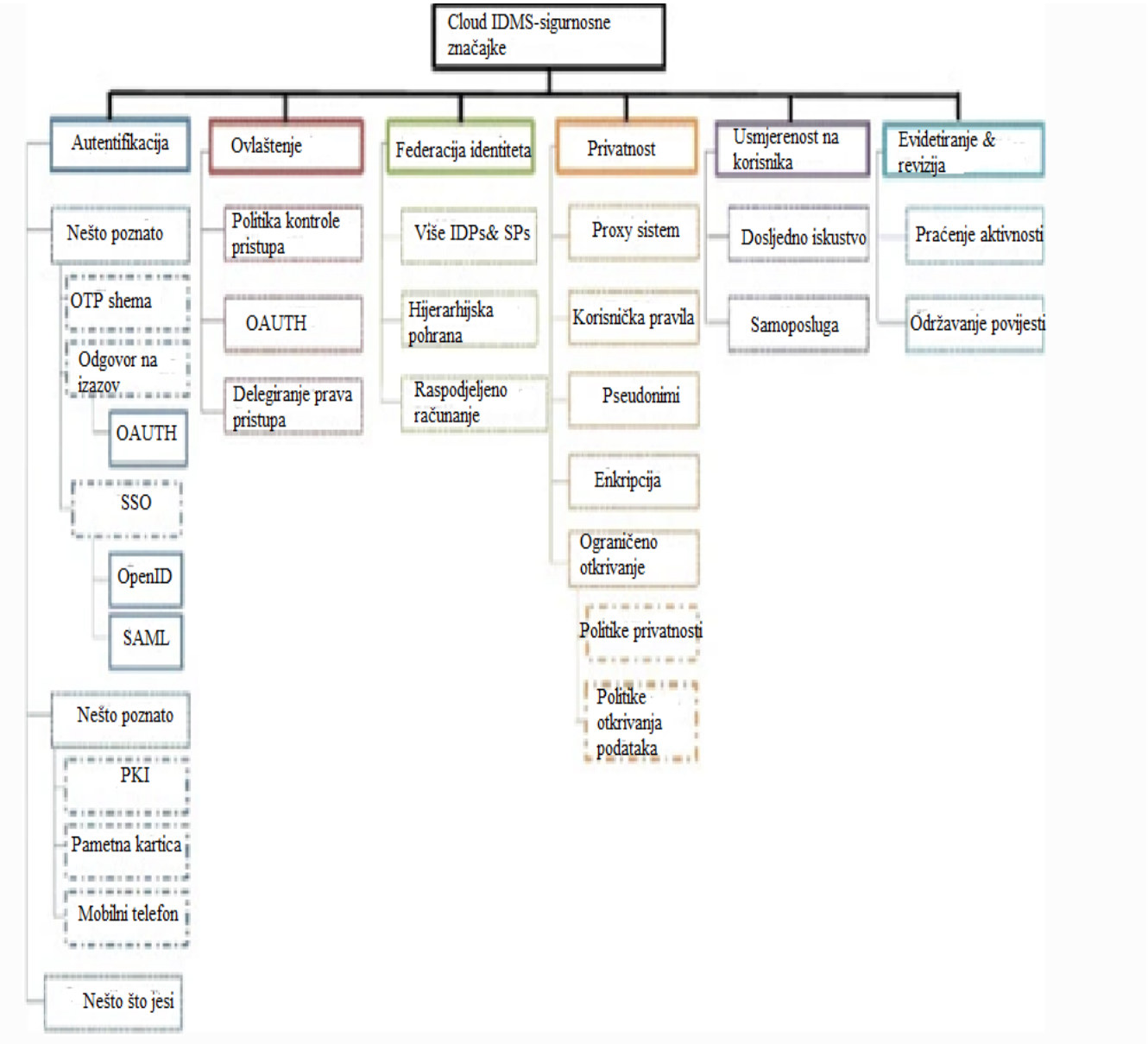
		modificira podatke koji se prenose između dva uređaja. Prisluškivanje, također poznato kao njuškanje ili njuškanje, oslanja se na nezaštićenu mrežnu komunikaciju za pristup podacima u prijenosu između uređaja.
A6	Postavljanje privilegija	Do postavljanja privilegija dolazi kada primjerice aplikacija dobije prava i privilegije koja joj ne bi trebala biti dostupna.
A7	Krađa identiteta	Neovlaštene osobe mijenjaju osobne dokumente izdane od strane upravnih tijela u zlonamjerne svrhe.
A8	Phising	Vrsta društvenog napada kojim se kradu korisnički podatci, kreditne kartice i sl.

5.2 Značajke i mehanizmi IDMS-a u oblaku: taksonomija

Cloud IDMS je jedan od najvažnijih značajki koje su potrebne za osiguranje sigurnosti CSC-ovih vjerodajnica. Većina identificiranih značajki postoji za tradicionalne IDMS-ove, ali kako Cloud donosi nove sigurnosne izazove, te moraju biti razmotrene ponovno iz perspektive Cloud okruženja. U tom smislu, Ferdous [12] je predstavio sveobuhvatnu taksonomiju koja pokriva gotovo sve funkcionalne i nefunkcionalne zahtjeve sustava za upravljanje identitetom, uključujući upotrebljivost, pouzdanost, pristupačnost, sigurnost i privatnost, kao što je prikazano na slici 5.1. Međutim, zbog svih ovih zahtjeva, sigurnost je jedan od ključnih čimbenika koji se kategorički smatra najvažnijim zahtjevom, a nije detaljno obrađen stoga zahtijeva daljnje istraživanje i istraživanje. Stoga je proširen njihov rad i pružena je dobro informirana taksonomiju (kao što je prikazano na slici 5.2) - koja temeljito pokriva sve potrebne sigurnosne značajke Cloud IDMS-a i njihove odgovarajuće mehanizme realizacije. Dok, za druge aspekte kao što su upotrebljivost, pristupačnost, odgovornost i provođenje zakona itd., potvrđujemo da je taksonomija predstavljena u Ferdous [12] pouzdana i primjerena, budući da uključuje sve ključne zahtjeve.

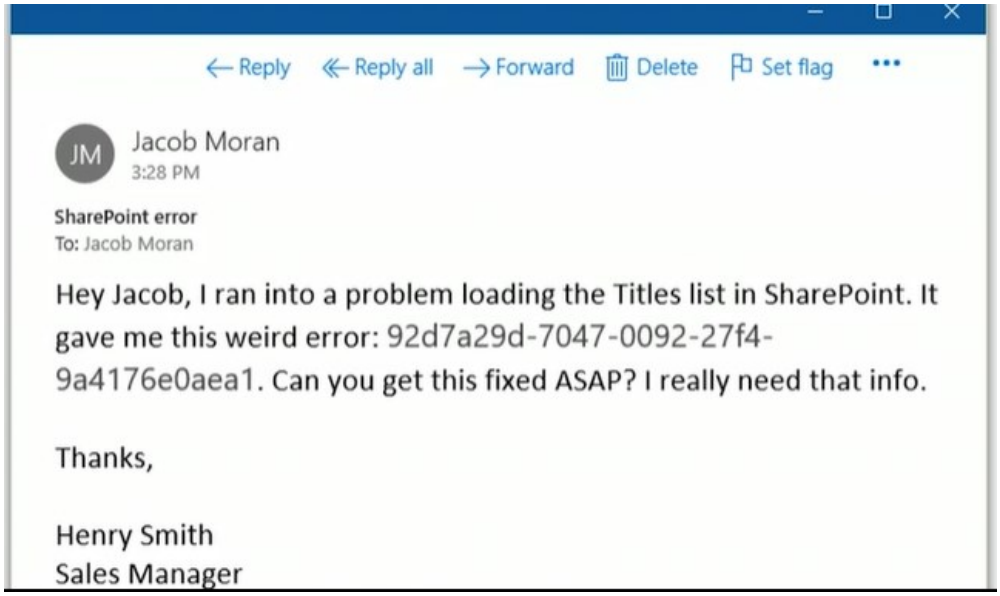


Slika 5.1 *Taksonomija zahtjeva za upravljanje identitetom*



Slika 5.2 *Taksonomsko stablo komponenti i mehanizama IDMS-a u oblaku*

5.3 Primjer sigurnosnog problema prilikom upravljanja digitalnim identitetom i njegovo rješenje

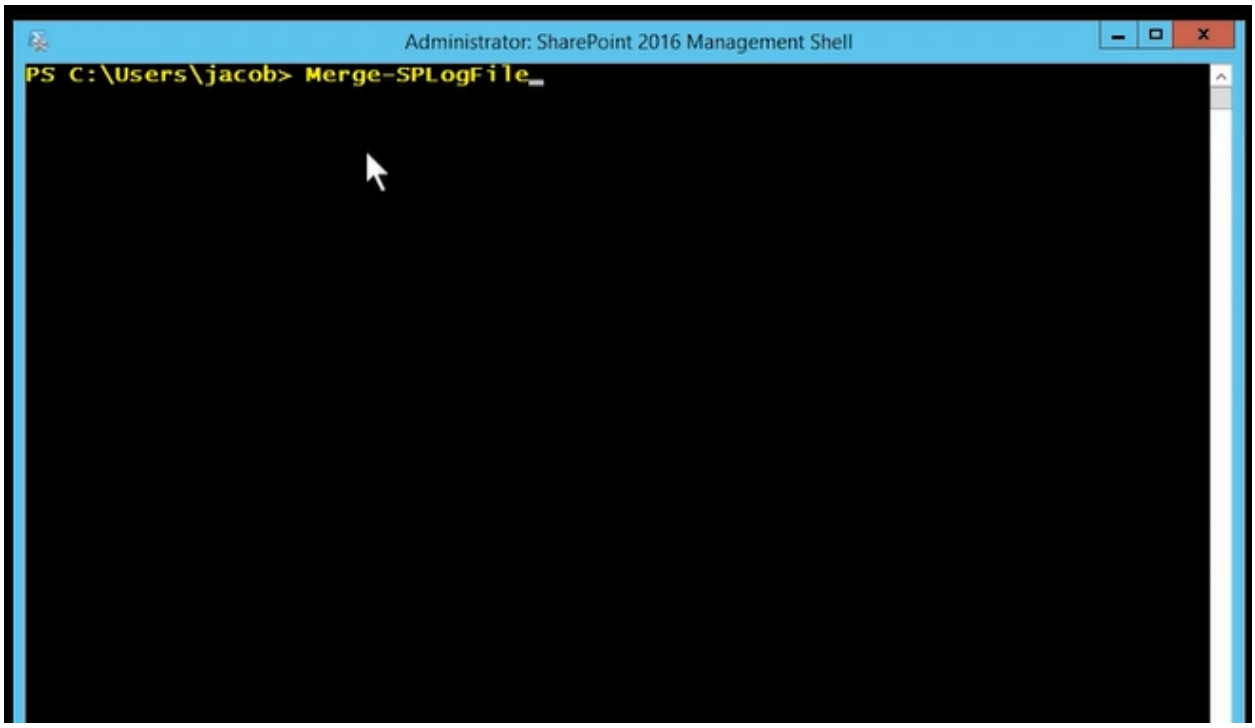


Slika 5.3 *Prikaz sigurnosnog problema*

Kako bi se riješio problem prva solucija je potražiti error koji se prikazuje. No kako to učiniti?

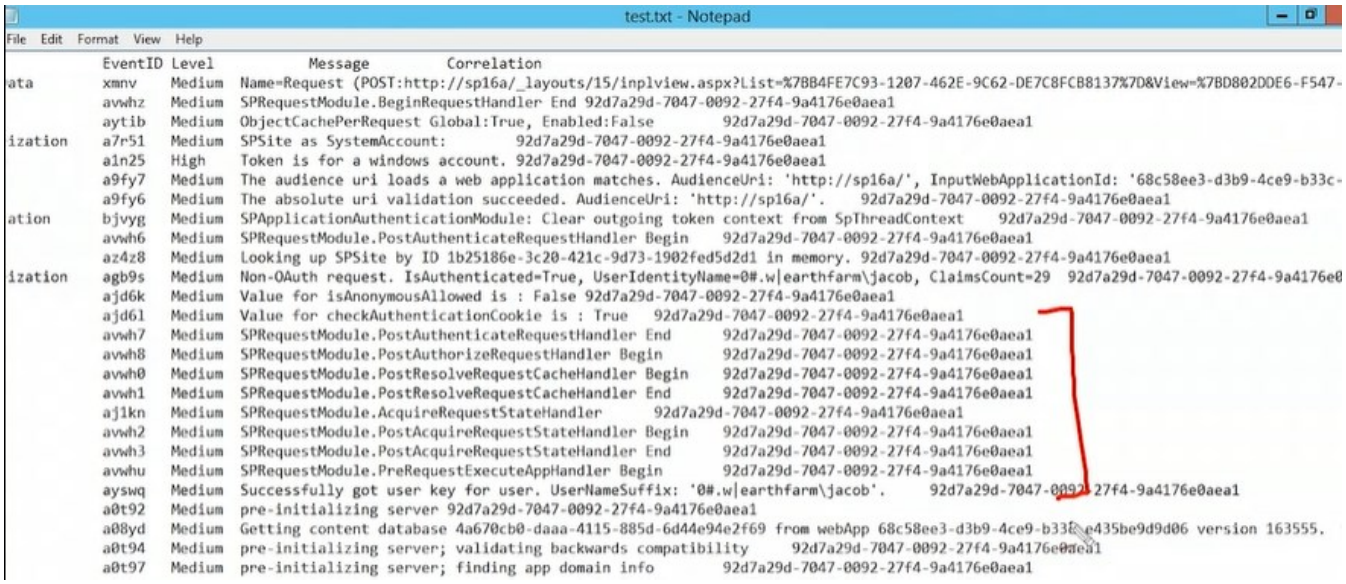
Kako bi se problem riješio mora se potražiti korelacijski ID. No važno je napomenuti kako korelacijski ID nije navedena pogreška. Nego jednostavno svaki put kada se javi problem pri dijeljenju zajedničke točke prvo se javlja opisani ID. Svaki proces je snimljen i dokumentiran. Ali drugo pitanje koje se postavlja je gdje se nalazi navedena dokumentacija? Dokumentacija se nalazi u *ULS trace log-u.* Koji se uobičajeno koriste za postavljene velike količine informacija.

Prvi korak koji bi se trebao napraviti je otvoriti *PowerShell* i upisati naredbu Merge-SPLogFile te se doda željena putanja [12].



Slika 5.4 *Prikaz upisivanja naredbe u PowerShell*

Ukoliko se otvori tekstualna datoteka mogu se vidjeti svi korelacijski ID-evi.



Slika 5.5 Prikaz korelacijski ID-eva u tekstualnoj datoteci

Kako bi se riješio problem moraju se riješiti pogreške vezane uz BSC. Stoga problem predstavljaju eksterni tipovi podataka. Korisnikov pristup eksternim tipovima podataka predstavlja problem ili možda eksterni tipovi podataka koji koriste bazu za pohranu ili njihovo spajanje na SQL i slično. Pretpostavka je da problem predstavlja SharePoint i njegovo povezivanje na SQL. Stoga se mora provjeriti radi li uopće SQL. Mora se provjeriti je li korisnik uopće povezan na bazu [12].

5.4 Problemi i incidenti koji se javljaju pri upravljanju digitalnim identitetima u oblaku te preporuke za njihovu prevenciju

Računarstvo u oblaku se odnosi na računarstvo putem interneta. Pri čemu se Internet ovdje odnosi na klaster oblaka. Računalni oblak svojim klijentima nudi virtualni oblak. *World Wide Web* je prebačen na jedno virtualno računalo pohranjeno na *Cloud.* Računalni oblak je širok pojam koji se ne može objasniti samo jednom riječju. Poprilično je unaprijeđen posljednjih nekolika godina te svoj razvoj seže u dalekoj prošlosti. Zbog veliko broja prednosti samog oblaka, pojavila su se i određena ograničenja. Najveću prepreku predstavlja sigurnost u oblaku i mogući incidenti vezani za nju. Kako bi korisnici ostali vjerni korištenju usluge na *Cloud-u,* mora se osigurati da su podatci pohranjeni na njemu na sigurnom. Korisnik i dalje ima pravo na sumnju, još uvijek ne može biti siguran jesu li njegovi podatci sigurno pohranjeni. No što ako podatcima pristupe neovlaštene osobe? Što ako neovlaštena osoba pristupi osobnim podatcima? Sva navedena pitanja predstavljaju problem korisnicima. Stoga se predlaže korištenje korisničkog identiteta za upravljanje podatcima spremljenim na oblaku. Na taj način će se postići sigurna udaljenost od sigurnosnih prijetnji. Upravljanje digitalnim identitetima u oblaku posjeduje i veliki broj prednosti [14]:

1. Pristup resursima: to je ujedno i najveća prednost računarstva u oblaku. Omogućuje pristup procesorskoj snazi više udaljenih računala. To klijentima omogućuje da iskoriste prednost veće brzine računanja i veći kapacitet pohrane od samog dijela cijene.

2. Mobilnost: korisnici mogu pristupi usluzi s bilo kojeg dijela svijeta budući da su usluge web-based i zbog pojave mobilnih uređaja. To zaposlenim ljudima olakšava pristup važnim poslovnim podatcima kada su u pokretu. Na primjer, zaposlenik može ispuniti određeni obrazac dok je u vlaku, programer riješiti određeni problem i sl, obavljajući ostatak posla sa pristupom tim podatcima u stvarnom vremenu.

3. Jednostavna skalabilnost: Sam aspekt skalabilnosti računarstva u oblaku ga čini prilično privlačnim za razvoj organizacija s različitim razinama potražnje računalnih resursa.

4. Sigurnost podataka i kapacitet pohrane: sigurnost je jako bitna jer ukoliko se dogodi određeni propust može uzrokovati ozbiljnu financijsku štetu. Stoga je i za očekivati da je u većini kompanija sigurnost podataka i kapacitet pohrane podataka koju nude podatkovni centri daleko bolja od one koju imaju obični građani.

5. Štednja: kod štednje nema plaćanja unaprijed kao što je bio ranije slučaj sa kupnjom licence u licenciranom ortodoksnom modelu softvera. Iako vjerojatno postoji početni korak ili naknada za konfiguraciju, to je obično poprilično rijetko. Pogodnosti sustava koju korisnici plaćaju prema korištenju organizacije sa rastom i padom su u potražnji sa stalnim računalnim resursima.

6. Podrška za održavanje: dobavljači će konstantno nuditi stalnu uslugu podrške. Međutim, udaljene hosting usluge čine proces održavanja i podrške usluge manje nametljive kupcima.

7. Ekološki prihvatljiv: javlja se prijedlog da podatkovni centri postanu zelena alternativa kućnom računarstvu. Razlog tomu je što poslužitelji u velikim podatkovnim centrima obično rade oko 80%, dok kućni poslužitelji mogu raditi s oko 5% kapaciteta. Vjerojatno je da je postojanje jeftinih i pristupačnih arhitektura za računarstvo u oblaku povećalo ukupnu potražnju za računanjem, koja premašuje učinkovite dobitke energije koji su ostvareni u podatkovnim centrima.

8. Besplatno probno razdoblje: neki dobavljači nude mogućnost isprobavanja proizvoda bez naknade. To je dobavljačima olakšalo povezanost s kupcima. Navedeni poslovni model se naziva *'free-mium'*.

Uz veliki broj prednosti postoje i brojni nedostatci kao što su[14]:

1. Internetska pouzdanost: It uslugama osiguranim putem interneta nedostaje pristup internetu ili imaju sporu vezu koja otežava pristup njihovim uslugama. To može predstavljati veliki problem u poslovanjima gdje su takve vrste usluge ključne za samo poslovanje. Sa sve većim razvitkom interneta ovo bi trebalo predstavljati manji problem. Međutim nigdje ne može postojati jamstvo neprekinute usluge čak i sa lokalnim serverom.

2. Ovisnost o dobavljaču: u računarstvu u oblaku korisnik ovisi o dobavljaču za svakodnevnim pristup IT uslugama. Ukoliko je dobavljač u financijskim problemima, njegova sposobnost izvršavanja usluge može biti upitna. Međutim, ovisnost o dobavljaču je karakteristična za većinu organizacija i uobičajena procjena rizika se može provoditi kako bi se smanjio rizik.

3. Sigurnost: kako bi zaštitili svoje podatke kompanije će morati imati odredbe unutar kojih se izravno specificira kako pružatelji usluge u oblaku planiraju zaštiti podatke.

4. Malo ili nimalo reference: zbog sigurnosnih briga, dobavljači u oblaku su nedostupni za većinu ili ne žele predstaviti studije slučajeva o tome koje kompanije trenutno koriste njihove usluge. Zapravo postoji mali broj kompanija koje javno izvještavaju o svojoj uporabi računarstva u oblaku. Zbog toga je veliki broj kompanija nesiguran u vezi korištenja resursa računarstva u oblaku iako je dosad sam pojam postao poprilično poznata terminologija. Drugi nedostatak je taj što mali broj kompanija koristi *Cloud* tehnologiju zbog postojećih problema. *Cloud* treba povezanost za svoj rad, ukoliko veza nije ostvarena, ne će biti moguć niti njegov rad. Čak i ako pružatelji usluge pažljivo prate vezu i dalje postoji rizik. Mogućnosti softvera su pak drug problem. Kada se radi o osobnim uređajima, bežično povezivanje može predstavljati problem. Problem može biti ukoliko postoji veliki broj podataka za obradu. Budući da nema konkretnog odgovara korisnika na to tko posjeduje podatke, on sam postavlja skup odredbi i uvjeta koje je ponekad teško

održavati. Veliki broj nedostataka samog računarstva postoji zbog činjenice da je tehnologija još relativno nova. Drugim riječima, problemi će se rješavati u hodu, što više i više korisnika bude prihvaćalo računarstvo u oblaku. Mogućnost pohrane velikog broja podataka je oduvijek predstavljala problem u IT-u, ali s tehnologijom koja zahtijeva informacije izvan virtualnih sigurnih zidova.

Sigurnost je jedan od glavnih čimbenika u računalstvu u oblaku[15]. Hakeri i zlonamjerni uljezi mogu provaliti u *Cloud* račune i ukrasti osjetljive podatke pohranjene u sustavima u oblaku .Metode pružanja sigurnosti koje se koriste su Kriptografija i sigurnosni algoritmi. Ali kako se pobrinuti da su informacije pohranjene na *Cloud-u* dovoljno sigurne da mu neovlaštene osobe ne mogu pristupiti .*Cloud* svojim korisnicima obećava povjerljivost, integritet i dostupnost. Sigurnost nije značajka; to je svojstvo sustava. Sva ta studija dovodi do upotrebe identiteta upravljanje koja je ranije objašnjena. Upravljanje identitetom bavi se identifikacijom kao što je sustav u nekoj zemlji. IDM sustavi su politike koje definiraju koji se uređaji koriste ili koja je mreža dopuštena . IDM u oblaku mora upravljati i kontrolirati virtualne uređaje, servisne identitete, kontrolne točke itd. IDM je važan dio oblaka. Sada pružatelji usluga u oblaku trebaju kontrolirati korisnička imena, lozinke i druge informacije koje se koriste za identifikaciju, provjeru autentičnosti i autorizirati korisnike za razne aplikacije. Primjeri: definicija pravila, izvješćivanje, upozorenja i alarmi. Neovlašteni korisnici pokušavaju se prijaviti na alarm i alarm ide dalje. Neki sustavi nude podršku za integraciju rječnika za žičane i bežične sustave. Modeli IDM i sigurnosni izazovi prikazani su u tablici 3.

Tablica 4. *Prikaz nedostataka i prednosti IDM modela*

IDM i SSO model	**Prednosti**	**Nedostatci**	**Sigurnosni izazovi**
Neovisni IDM stog	-jednostavni za implementaciju -nema zasebne integracije s imenikom poduzeća	-korisnici moraju pamtiti odvojene vjerodajnice za svaku *Saas* aplikaciju	-idm stog bi trebao biti visoko konfigurabilan kako bi se olakšala usklađenost s poslovnim pravilima poduzeća, primjerice jačina lozinke i sl.
Sinkronizacija vjerodajnica	-korisnici ne moraju pamtiti višestruke lozinke	-zahtijeva izravnu integraciju s poduzećem -postoji veći sigurnosni rizik zbog prijenosa vjerodajnica izvan parametara poduzeća	-Saas prodavači trebaju osigurati sigurnost vjerodajnica prilikom prijenosa i pohrane te spriječiti njihovo curenje
Federacijski IDM	-korisnici ne moraju pamtiti višestruke lozinke -nema zasebne integracije s imenikom poduzeća -niske sigurnosne vrijednosti u usporedbi sa sinkronizacijom vjerodajnica	-relativno su kompleksnije za implementaciju	- Saas prodavači trebaju osigurati prikladnu povjerljivu vezu kako bi se osigurala federacija identiteta korisnika

Upravljanje identitetom ne uključuje samo predstavljanje samoga sebe, već i odgovor na niz prijetnji koje proizlaze iz širenja osobnih podataka na internetu. Navedena su dva događaja vezana za krađu identiteta i povredu podataka u glavnim uslugama poslužitelja na *Cloud-u:*

- *Apple*-nedavno je *Apple* doživio nekoliko loših trenutaka. Najviše glasina je bilo za hakiranje fotografija poznatih osoba, koje su se na kraju pokazale istinitima. Hakiranje se provelo sa *iCloud-a* i to baš u vrijeme lansiranja najnovijih *Applevih* telefona. Kupci za napad okrivljavaju kompaniju, dok *Apple* tvrdi da je za napad kriva gruba sila. Napad grubom silom je kada haker pokušava otkriti sve moguće lozinke dok ne uspije. Prvo je pogodio one sa najslabijim lozinkama. Ovaj pad je poprilično utjecao na Apple prodaju, ali oni i dalje tvrde da je odgovornost kupaca na njima samima da brinu o svojim osobnim stvarima na internetu.

- *Yahoo*-bilo je nekoliko slučajeva krađe email računa. Lozinke i korisnička imena su prikupljena od baze podataka koje su bile u vlasništvu trećih strana koje su bili kompromitirane. Tvrtka je tada slala obavijest svim svojim korisnicima da promijene svoje postojeće lozinke u nove. Hakirani kupci su dobili obavijest tvrtke o njihovom gubitku. Na pohranjene podatke će se primijeniti različite metode šifriranja u slučaju da im neovlašteni korisnici ne mogu pristupiti. Ukoliko osoba ne uspije proći sigurnosnu provjeru, on ili ona neće dobiti ovlast za korištenje podataka pohranjenih na oblaku. Tajni podatci ne će biti dostupni korisniku čak i ako prođe sigurnosna pitanja, zbog izrazito učinkovite sigurnosne provjere.

5.4.1 Rješenja za sigurnosne probleme

Potrebno je osigurati sigurnost na različitim razinama prijave korisnika[16]:

- Klasificiranje podataka prema razinama sigurnosti i primjena različitih shema šifriranja za zaštitu klasificiranih podataka.
- Osvježavanje ključeva nakon određenog vremenskog razdoblja.

Podatci koji će biti spremljeni na *Cloud-u* će biti sustavno raspoređeni prema njihovoj osjetljivosti. To zapravo znači da podatci koji trebaju veliku razinu sigurnosti će biti spremljeni na visokoj razini sigurnosti. Podatci poput primjerice lozinke švicarskog računa mogu biti zaštićeni Z sigurnošću. Korisnikova potreba da se prijavi na svoj račun, na kojemu su podatci pohranjeni mora zadovoljavati određena sigurnosna pitanja. Mora upisati svoju lozinku za pristup podatcima. Postojati će veliki broj sigurnosnih provjera na ovoj razini kako ne bi važne informacije dospjele u loše ruke. Podatci sa srednjom osjetljivošću će biti spremljeni na mediju kontrole sigurnosti. Kao što su primjerice podatci o svim bankovnim računima i novcima pohranjenih na njemu,. Sigurnosna provjera ovdje su sigurnosna pitanja na koje će korisnik morati odgovoriti kako bi dobio pristup svojim pohranjenim podatcima. Oni koji su ostali s najmanjom zaštitom podataka će obično biti osigurani korisničkim *ID-em* i lozinkom za prijavu.

Ishod predloženih rješenja je sustav koji pruža sigurnost u oblaku, tj sigurnost podataka pohranjenih na njemu uz pomoć sustava upravljanja identitetom. Uneseni podatci su osigurani uz pomoć raznolikih algoritama enkripcije i korištenja identiteta pojedinaca. Sigurnosne provjere podržavaju svaku razinu na kojoj podatci trebaju biti pohranjeni. Razina povjerljivosti pomaže da podatci budu zaštićeni od hakera uz pomoć skeniranja otisaka prstiju i sigurnosnih pitanja kojih ne će biti tako lako riješiti. Druga razina je za internu uporabu, oni podatci kojima je potrebna srednja razina sigurnosti. Takva sigurnost je pružena uz pomoć sigurnosnih pitanja koja nisu ista kao ona za najvišu razinu sigurnosti ili kao ona na razini jedan.

Treća vrsta sigurnosne provjere je jednaka kao i na svakoj društvenoj mreži. Korisnik ispuni svoje korisničko ime i lozinku i možda dobije pristup svojim podatcima. Glavni razlog koji čini ovaj sustav drugačijim od ostalih je korištenje identiteta osobe. To je objašnjeno korištenjem skeniranja otisaka prstiju koje nisu sklone napadima. Druga vrste sigurnosti pružena ovim sustavom je osvježavanje ključeva ili lozinki nakon određenog vremenskog razdoblja. Na taj način se sustav štiti od napada poput primjerice napada grubom silom.

6 .ZAKLJUČAK

Kao što je ranije navedeno u ovom diplomskom radu, računarstvo u oblak ,tehnologije su brzo usvojene od strane organizacije kako bi smanjili troškove i omogućili fleksibilnu i učinkovit pristup kritičnim podacima. Od novih tehnologija koje se pojavljuju u oblaku su kibernetička sigurnost i izazovi povezani s tim tehnologijama te su se povećali velikom brzinom. Uloženi su neki napori kako bi se riješili navedeni sigurnosni problemi u oblaku. U ovom radu raspravljeno je o identitetu upravljanja razvoja uključujući razne koncepte poput jedinstvene prijave te i otvorene, o čemu se raspravljalo u sigurnosnim pitanjima za oblak. Zatim je raspravljeno o nekim izazovima koji se javljaju prilikom upravljanja digitalnim identitetima u oblaku. Posebno je pažnja usmjerena na autentifikaciju, autorizaciju i kontrolu pristupa, reviziju i odgovornost te povjerenje i povjerljivost pitanja za *Cloud.* Također je raspravljeno o ciljevima novoosnovanog tehničkog povjerenstva OASIS-A za upravljanje digitalnim identitetom. Što je veći napredak ostvaren koristeći opisani odbor i sa što većim brojem provedenih istraživanja mogu se očekivati obećavajuća rješenja za svaki slučaj sigurnosnog problema prilikom upravljanja digitalnim identitetima. Postoji i određen broj nedostataka i sigurnosnih problema koji se javljaju pri upravljanju digitalnim identitetima u oblaku no predložen je niz sigurnosnih rješenja koji pružaju korisnicima određenu razinu zaštite. Budući da je danas poprilično nemoguće jamčiti sigurnost korisnika na internetu sa 100% sigurnošću iako je tehnologija poprilično napredovala.

LITERATURA

[1]MaryLine, Digital Identity Manangment: 2015, str 300

[2]K. Hamlen et al, Security Issues for Cloud Computing, Journal of Information Security and Privacy, 2010, str 150-30

[3]IDENTITY Management for Cloud Computing: Developments and directions. Preuzeto sa: https://personal.utdallas.edu/~hamlen/hamlen11csiirw.pdf

[4]Moretti, C., Steinhaeuser, K., Thainer, D., & Chawla, N. (2008). Scaling Up Classifiers to Cloud Computers. In Proceedings of the IEEE ICDM

[5]Hadoop. (n.d.). Preuzeto sa http://hadoop.apache.org

[6]Teswanich, W., & Chittayasothorn, S. (2007). A Transformation of RDF Documents and Schemas to Relational Databases. IEEE Pacific Rim Conferences on Communications, Computers, and Signal Processing, 38-41

[7] Security problems. Preuzeto sa Percipio: https://atos.percipio.com/courses/1af201a0-0967-11e9-8325-1de0a2cbe9ba/videos/4a234e20-0967-11e9-8325-1de0a2cbe9ba?tab=overview

[8]Kerberos, preuzeto sa : https://www.techtarget.com/searchsecurity/definition/Kerberos

[9]M. Choudhury et al, SemID: Combining Semantics with Identity Management, SECUREWARE, 2009/

[10]G. J. Walters, "Privacy and Security: An Ethical Analysis", Computers and Society, 2001, pp. 8-23.

[11] Standards for digital identity. Preuzeto sa : http://www.oasisopen.org/committees/tc_home.p hp?wg_abbrev=idcloud

[12]Cloud identity management security issues & solutions: a taxonomy https://casmodeling.springeropen.com/articles/10.1186/s40294-014-0005-9

[13] Configuration of KDC and Client. Preuzeto sa Percipio: https://atos.percipio.com/courses/102bbb60-7931-11e8-964e-a1eb008266c4/videos/2a1b22e0-7931-11e8-964e-a1eb008266c4

[14] Farhan Bashir Shaikh and Sajad Haider, “Security Threats in Cloud Computing“, IEEE, sixth International Conference on Internet Technology and Secured Transactions, December,2011

[15] D.Barnard- Wills and D.Ashenden, “Public sector engagement with online identity menagement“, Springer Cranfield University, Shrivenham, Swindon SN6 8LA,UK.

[16] “Data classification for Cloud readiness“, Microsoft Thrustworthy Computing

SAŽETAK

U ovom diplomskom radu detaljno je opisano upravljanje digitalnim identitetima u oblaku. U uvodu je nešto detaljnije rečeno o tehnologijama u oblaku. Nakon toga su definirani koncepti računalstva u oblaku, principi koji se koriste, modeli, svojstva i ostale karakteristike. Objašnjeno je sigurno računalstvo u oblaku, sigurni hipervizori i šifriranje podataka. Zadnja dva poglavlja objašnjavaju digitalno upravljanje u oblaku te standarde i nedostatke koji se javljaju prilikom upravljanja. Naveden je detaljan primjer sigurnosnog problema i prijedlog njegovog rješenja.

Ključne riječi: Cloud, sigurnost i digitalni identitet.

ABSTRACT

This thesis describes in detail the management of digital identities in the cloud. In the introduction, something was said in more detail about cloud technologies. After that, the concepts of cloud computing, the principles used, models, properties and other characteristics are defined. Secure cloud computing, secure hypervisors and data encryption explained. The last two chapters explain digital management in the cloud and the standards and shortcomings that arise during management. A detailed example of a security problem and a proposal for its solution are given. The detailed example of security problem is given and solution for it.

Key words: digital identity, security,Cloud.