UNIVERZITET U NIŠU
ELEKTRONSKI FAKULTET
Katedra za računarstvo

APACHE CASSANDRA - SIGURNOST BAZE PODATAKA

Predmet: Sistemi za upravljanje bazama podataka
-Seminarski rad-

Student:

Marijana Cvetković, br. ind. 1431

Mentor:

Doc. dr Aleksandar Stanimirović

Niš, jun 2022. godina

Sadržaj

1. UVOD

Baza podataka se može definisati kao organizovani skup podataka spremljen na tvrdom disku računara. Omogućuje svakom korisniku brz i lagan pristup, unos i analizu podataka. Uzimajući u obzir značaj podataka unutar kompanije ili organizacije, neophodno je osigurati podatake unutar njene baze podataka.

Sigurnost baze podataka obuhvata metode zaštite baze podataka od neovlašćenog pristupa, modifikacija ili uništenja podataka. Uz zaštitu podataka, mora se voditi računa i o privatnosti pojedinaca čije se podatke sprema. Bezbednosni programi baze podataka su dizajnirani da zaštite ne samo podatke u bazi podataka, već i sam sistem za upravljanje podacima i svaku aplikaciju koja im pristupa od zloupotrebe, oštećenja i upada.

Cilj ovog rada je da detaljno opiše mere sigurnosti baza podataka uopšte, a zatim i detaljno mere bezbednosti Cassandra baze podataka i pregled sigurnosnih metoda koji se sprečavaju različiti upadi, krađa podataka, neadekvatno korišćenje podataka i ostalo.

U drugom poglavlju se nalazi se opis pretnji sa kojima se baze podataka suočavaju, zatim kako se server baze podataka obezbeđuje, saveti za bezbedno korišćenje i očuvanje podataka i na kraju politike i kontrole koje se primenjuju nad bazama podataka.

Treće poglavlje sadrži sve o sigurnosti Apache Cassandra bazi podataka, koje su to osnovne funkcije za očuvanje, način na koji se obezbeđuju autentifikacija i autorizacija, opis SSL šifrovanja podataka za bezbedan prenos podataka i opšte mere bezebednosti.

Cassandra obezbeđuje bezbednu komunikaciju između klijentske mašine i klastera baze podataka i između čvorova unutar klastera. Omogućavanje šifrovanja osigurava da podaci u letu nisu ugroženi i da se bezbedno prenose. Opcije za šifrovanje od klijenta do čvora i od čvora se upravljaju odvojeno i mogu se nezavisno konfigurisati. Za informacije o generisanju datoteka skladišta ključeva i skladišta poverenja potrebnih sa Java podržanim kladištima ključeva koja se koriste u SSL komunikciji. Počevši od Cassandre 4, Cassandra podržava ponovno učitavanje SSL sertifikata. SSL sertifikat je deo koda navišem veb serveru koji obezbeđuje sigurnost za onlajn komunikaciju.

Četvrto poglavlje sadrži opis baze podataka, podataka koje sadrži i načina na koji se kreira. Nad tim skupom podataka, primenjene su mere bezbednosti kako bi se osigurali podaci od različitih upada i nekontrolisanog korišćenja. Kreirani su razčičiti tipovi korisnika koji imaju različite privilegije pristupa i korišćenja baze podataka. Takođe, svaki od korisnika ima svoje korisničko ime i lozinku za pristup bazi podataka.

U petom poglavlju dat je zaključak izveden na osnovu svih mogućih dostupnih informacija o bezbednosti Cassandra baze podataka, kao i praktične primene nad postojećom bazom podatka.

2. SIGURNOST BAZA PODATAKA

Bezbednost baze podataka se odnosi na kolektivne mere koje se koriste za zaštitu i obezbeđenje baze podataka ili softvera za upravljanje bazom podataka od nezakonite upotrebe i zlonamernih sajber pretnji i napada. Sigurnosne procedure baze podataka imaju za cilj zaštitu ne samo podataka unutar baze podataka, već i sistema upravljanja bazom podataka i svih aplikacija koje joj pristupaju od upada, zloupotrebe podataka i oštećenja. To je širok pojam koji uključuje mnoštvo procesa, alata i metodologija koje osguravaju sigurnost u okruženju baze podataka. [1]

Bezbednost baze podataka pokriva i sprovodi bezbednost na svim aspektima i komponentama baze podataka. Ovo uključuje:

- Podaci pohranjeni u bazi podataka,
- Server baze podataka,
- Sistem za upravljanje bazom podataka (DBMS),
- Druge aplikacije toka rada baze podataka.

Bezbednost baze podataka generalno planira, implementira i održava administrator baze podataka ili drugi stručnjak za bezbednost informacija. Neki od načina na koje se analizira i implementira bezbednost baze podataka uključuju:

- Ograničavanje neovlašćenog pristupa i korišćenja primenom jakih i višefaktorskih kontrola pristupa i upravljanje podacima.
- Testiranje opterećenja/stres i testiranje kapaciteta baze podataka kako bi se osiguralo da se ne sruši u napadu distribuiranog uskraćivanja usluge ili preopterećenju korisnika.
- Fitička sigurnost servera baze podataka i rezervne opreme od krađe i elementarnih nepogoda. Redovne rezervne kopije podataka mogu se planirati kao deo bezbednosnog protokola baze podataka, a više kopija se može uskladištiti van lokacije da bi se obezbedila redudantnost i hitan oproravak.
- Pregled postojećeg sistema za sve poznate ili nepoznate ranjivosti i definisanje i implementacija mape puta/plan za njihovo ublažavanje.
- Šifrovanje podataka može da obezbedi nivo bezbednosti za zaštiti integiteta i poverljivosti podataka.

Sprovođenje adekvatnih bezbednosnih praksi baze podataka je od vitalnog značaja za svaku organizaciju iz različitih razloga. Ovo uključuje:

- **Obezbeđivanje kontinuiteta poslovanja**: Mnoga preduzeća ne mogu da rade dok se kršenje ne reši.
- **Minimiziranje finansijske štete**: Jednom kada dođe do kršenja, organizacija mora da izdrži značajne finansijske troškove kako bi saopštila krešenje svim svojim klijentima, upravljala krizom, popravila ili ažurirala pogođene sisteme i hardver, platila istražne aktivnosti itd.
- **Gubitak intelektulane svojine**: Ako se pristupi bazi podataka, postoji šansa da su poslovne tajne kompanije, vlasničke procedure i drugi oblici intelektualne svojine ukradeni illi razotkriveni. U nekim slučajevima, to znali potpuni gubitak svake konkurentske prednosti koju ta organizacija održava.
- **Šteta reputaciji brenda**: Kako se baza korisnika obavesti o kršenju, partneri i kupci mogu

izgubiti veru u sposobnost organizacije da zaštiti svoje podatke. Reputacija brenda će pamtiti i mnogi će možda odlučiti da više ne kupuju proizvode ili usluge te organizacije.

- **Kazne i novčane kazne**: Organizacije moraju da budu u skladu sa velikim brojem propisa, kao što su oni u Opštoj naredbi o zaštiti podataka, Standardu bezbednosti podataka industrije platnih kartica, Zakonu o prenosivosti i odgovornosti zdravstvenog osigurannja i više. Ako dođe do povrede podataka zato što organizacija nije ispoštovala ove propise, novčane kazne i kazne mogu biti veoma ozbiljne, u nekim slučajevima čak i premašiti nekoliko miliona dolara po kršenju.

2.1. Sigurnosni ciljevi

Sigurnosni ciljevi su definisani po CIA (engl. Confidentiality Integrity Availability) modelu. Kršenje ijedne stavke CIA modela baza podataka postaje nesigurna.

Gubitak poverljivosti – poverljivost baze podataka se odnosi na zaštitu podataka od nedozvoljenog razotkrivanja. Rezultat nedozvoljenog razotkrivanja može biti u sporu od kršenja zakona o zaštiti svojih podataka do nacionalne sigurnosti. Takvi propusti u sigurnosti mogu dovesti do gubitka poverenja javnosti ili čak sudskog postupka protiv organizacije.

Gubitak integiteta – itegritet baze podataka se odnosi na zahtev da su podaci zaštićeni od neprikladnih izmena. Izmene podataka uključuju stvaranje, umetanja, ažuriranje, menjanje stanja podataka i brisanje. Integritet se gubi ako dođe do neodobrenih izmena. Ako se gubitak sastava ili integritet podataka ne ispravi, daljnja upotreba sastava može dovesti do netačnosti, pogrešnih odluka ili prevare (ako je neko netačno izmenio podatke).

Gubitak raspoloživosti – raspoloživost baze podataka se odnosi na pružanje usluge i podataka osobi ili programu koji imaju prava na nju. Ponekad zvan uskraćivanje usluge. Kada baza podataka nije dostupna nastaje gubitak, pa se svaka pretnja koja povećava vreme tokom kojeg baza podataka nedostupna izbegava. [2]

2.2. Pretnje bezbednosti baze podataka

Mnoge ranjivosti softvera, pogrešne konfiguracije ili obrasci zloupotrebe ili nepažnje mogu dovesti do kršenja. Evo nekoliko najpoznatijih uzroka i tipova sajber pretnji bezbednosti baze podataka:

- Insajderske pretnje

Insajderska pretnja je bezbednosni rizik iz jednog od sledećih tri izvora, od kojih svaki ima privilegovani način ulaska u bazu podataka: zlonamerni insajder sa lošom namerom, nemerna osoba unutar organizacije koja izlaže bazu podataka napadu nepažljivim radnjama i autsajder koji dobija akreditive putem društvenog inženjeringa ili drugih metoda, ili dobija pristup akreditivima baze podataka.

- Ljudska greška

Slabe lozinke, deljenje lozinki, slučajno brisanje ili oštećenje podataka i druga nepoželjna

ponašanja korisnika i dalje su uzrok skoro polovine prijavljenih povreda podataka.

- Iskorišćavanje ranjivosti softvera baze podataka

Napadači stalno pokušavaju da izoluju i ciljnu ranjivost u softveru, a softver za upravljanje bazom podataka je veoma vredna meta. Nove ranjivosti se otkrivaju svakodnevno, a sve platforme za upravljanje bazama podataka otvorenog koda i komercijalni proizvođači softvera za baze podataka redovno izdaju sigurnosne zakrpe. Međutim, ako se zakrpe ne koriste brzo, baza podataka može biti izložena napadu. Čak i ako se na vreme primene zakrpe, uvek postoji rizik od napada nultog dana, kada napadači otkriju ranjivost, ali ona još uvek nije otkrivena i zakrpljena od strane dobavljača baze podataka.

- SQL/NoSQL injekcioni napadi

Pretnja specifična za bazu podataka uključuje upotrebu proizvoljnih ne-SQL i SQL stringova napada u upitima baze podataka. Obično su to upiti kreirani kao proširenje obrazaca za veb aplikacije ili primljeni putm HTTP zahteva. Svaki sistem baze podataka je ranjiv na ove napade, ako se programeri ne pridržavaju praksi bezbednog kodiranja i ako organizacija ne sprovodi redovno testiranje ranjivosti.

- Napadi prekoračenja bafera

Prelivanje bafera se dešava kada proces pokušava da upiše veliku količinu podataka u blok memorije fiksne dužine, više nego što je dozvoljeno da zadrži. Napadači mogu koristiti višak podataka, koji se čuvaju na susednim memorijskim adresama, kao početnu tačku za pokretanje napada.

- Napadi uskraćivanja usluge

U napadu uskraćivanja usluge, sajber kriminalac nadvaladava ciljnu uslugu – u ovom slučaju server baze podataka – koristeći veliku količinu lažnih zahteva. Rezultat je da server ne može da izvrši prave zahteve stvarnih korisnika i često se ruši ili postaje nestabilan. U distribuiranom napadu uskraćivanja usluge, lažni saobraćaj generiše veliki broj računara, koji učestvuju u botnetu koji kontroliše napadač. Ovo generiše veoma velike količine saobraćaja, koje je teško zaustaviti bez visoko skalabilne odbrambene arhitekture.

- Zlonamernih progarama

Malver je softver napisan da iskoristi ranjivosti ili da nanese štetu bazi podataka. Zlonamerni softver može da stigne preko bilo kog krajnjeg uređaja povezanog na mrežu baze podataka. Zaštita od zlonamernog softvera je važna na bilo kojoj kranjoj tački, a posebno na serverima baza podataka, zbog njihove visoke vrednosti i osetljivosti.

- Razvojno IT okruženje

IT okruženje koje se razvija čini baze podataka podložnijim pretnjama. Evo trendova koji mogu dovesti do novih vrsta napada na baze podataka ili mogu zahtevati nove odbramene mere:

  - Rastuća količina podataka – skladištenje, prikupljanje i obrada podataka eksponencijalno raste u skoro svim organizacijama. Sve prakse ili alati za bezbednost podataka moraju biti visoko skalabilni da bi odgovorili na daleke i bliske buduće zahteve.
  - Distribuirana infrastruktura – mrežna okruženja postaju sve složenija, posebno kako

preduzeća prenose radna opterećenja u hibridni oblak ili arhitekture sa više oblaka, što otežava primenu, upravljanje i izbor bezbednosnih rešenja.

- Sve strožiji regulatnorni zahtevi – svetski pejzaž usklađenosti sa propisima postaje sve složeniji, tako da praćenje svih mandata postaje sve izazovnije.
- Nedostatak veština za sajber bezbednost – postoji globalni nedostatak kvalifikovanih stručnjaka za sajber bezbednost, a organizacijama je teško da popune bezbednosne uloge. Ovo može otežati odbranu kritične infrastrukture, uključujući baze podataka. [3]

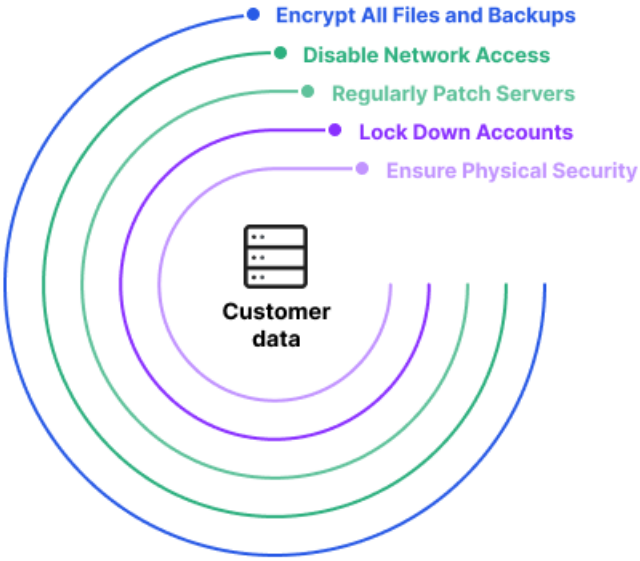
Pretnje se mogu podeliti u namerne ili nenamerne. Nenamerne pretnje su one koje su nastale slučajno, bez namere da se nanese zlo za organizaciju. Neke od njih su kada korsnik može nenamerno napraviti zahtev za objektom ili operacijom za koju ne bi trebao imati odobrenje, ali se zahtev odobrava zboh propusta u dodeli uloga; greška u komunikacijskom sastavu može spojiti korisnika u sednicu koja pripada drugom korisniku s drugačijim pravima pristupa; osoba može slučajno primiti poruku koja je namenjena drugom korisniku što može dovesti do gubitka poverljivosti u bazi podataka; operacijski sastav može slučajno prepisati preko podataka i uništiti deo baze podataka, dohvatiti pogrešne podatke i onda nesvesno poslati korisniku ili zakazati u brisanju podataka koje je trebalo uništiti.

Namerne sigurnosne pretnje nastaju kada korisnik namerno dobije neovlašćeni pristup i izvodi neovlašćene operacije nad bazom podataka. Takvi napadi se izvode s ciljem prouzrokovanja štete organizaciji. Primeri nekih su: nezadovoljni zaposleni upoznat sa organizacijskom bazom podataka želi nauditi organizaciji; industrijski špijun pokušava pristupiti podacima za konkurenciju; ovlašćeni korisnici poput administratora baze podataka pristupaju privatnim podacima krajnjih korisnika za koje ne bi trebali imati pristupa. [2]

2.3. Kako obezbediti server baze podataka

Server baze podataka je fizička ili virtuelna mašina koja pokreće bazu podataka. Obezbeđivanje servera baze podataka, takođe poznato kao 'očvršćavanje', je proces koji uključuje fizičku bezbednost, bezbednost mreže i bezbednu konfiguraciju operativnog sistema.

Na slici 1 su dati nivoi bezbednosti koje treba postići:



Slika 1: Nivoi bezbednosti

- **Obezbediti fizičku bezbednost baze podataka**

Potrebno je uzdržati se od deljenja servera za veb aplikacije i aplikacije baze podataka ako baza podataka sadrži osetljive podatke. Iako bi moglo biti jeftinije i lakše da se zajedno hostuju veb lokacija i baza podataka kod provajdera hostinga, sigurnost podataka se stavlja u tuđe ruke. Ukoliko se oslanja na uslogu veb hostinga za upravljanje bazom podatka, trebalo bi uveriti se da je to kompanija sa jakim bezbednosnim iskustvom. Najbolje je kloniti se besplatnih usluga hostinga zbog mogućeg nedostatka sigurnosti. Ako se vrši upravljanje bazom podataka u lokalnom data centru, potrebno je imati na umu da je centar podataka takođe podložan napadima sa strane ili insajderskim pretanjama. Neophodno je uveriti se da postoje fizičke mere bezbednosti, uključujući brave, kamere i bezbednosno osoblje u fizičkom objektu. Svaki pristup fizičkim serverima mora biti evidentiran i odobren samo ovlašćenim pojedincima. Pored toga, ne treba ostavljati rezervne kopije baze podataka na lokacijama koje su javno dostupne, kao što su privremene particije, veb fascikle ili neobezbeđene kante za skladištenje u oblaku.

- **Zaključati naloge i privilegije**

Biće objašnjeno na primeru Oracle baze podataka. Nakon što se baza podataka instalira, Oracle pomoćnik za konfiguraciju baze podataka automatski ističe i zaključava većinu podrazumevnih korisničkih naloga baze podataka. Ukoliko se ručno instalira Oracle baza podataka, to se neće desiti i podrazumevani privilegovani nalozi neće biti istekli ili zaključani. Njihova lozinka ostaje ista kao i korisničko ime, podrazumevano. Napadač će prvo pokušati da iskoristi ove akreditive da se poveže sa bazom podataka. Važno je osigurati da svaki privilegovani nalog na serveru baze podataka bude konfigurisan sa jakom, jedinstvenom lozinkom. Ako nalozi nisu potrebni, treba ih isteći i zaključati. Za preostale naloge, pristup mora biti ograničen na apsolutni minimum koji je potreban. Svaki nalog treba da ima pristup samo tabelama i operacijama (na primer, SELECT ili INSERT) koji zahteva korisnik. Izbegavati kreiranje korisničkih naloga sa pristupom svakoj tabeli u bazi podataka.

- **Redovno zakrpiti serevere baze podataka**

Potrebno je uveriti da zakrpe ostaju aktuelne. Efikasno upravljanje zakrpama baze podataka je ključna bezbednosna praksa jer napadači aktivno traže nove bezbednosne propuste u bazama podataka, a novi virusi i malver se pojavljaju svakodnevno. Pravovremena primena ažuriranih verzija servisnih paketa baze podataka, kritičnih bezbednosnih hitnih ispravki i kumulativnih ažuriranja poboljšaće stabilnost performansi baze podataka.

- **Onemogućiti pristup javnoj mreži**

Organizacije čuvaju svoje aplikacije u bazama podataka. U većini scenarija iz stvarnog sveta, krajnjem korisniku nije potreban direktan pristup bazi podataka. Stoga bi trebalo da se blokira sav pristup javnoj mreži serverima baze podataka. U idealnom slučaju, organizacija bi trebalo da postavi servere mrežnog prolaza za udaljene administratore.

- **Šifrovanje svih datoteka i rezervnih kopija**

Bez obzira na to koliko je odbrana čvrsta, uvek postoji mogućnost da se haker infiltrira u sistem. Ipak, napadači nisu jedina pretnja bezbednosti baze podataka. Zaposleni takođe mogu predstavljati rizik za poslovanje. Uvek postoji mogućnost da će zlonamerni ili nemarni insajder dobiti pristup datoteci kojoj nema dozvolu za pristup. [3]

2.4. Saveti očuvanja bezbednosti baze podatka

U nastavku će biti dati saveti koji se mogu koristiti za poboljšanje bezbednosti osetljivih baza podataka:

- **Aktivno upravljanje lozinkama i korisničkim pristupom**

Ukoliko je organizacija velika, potrebno je razmišljati o automatizaciji upravljanja pristupa putem upravljanja lozinkama ili softvera za upravljanje pristupom. Ovo će omogućiti dozvoljenim korisnicima kratkoročnu lozinku sa pravima koja su im potrebna svaki put kada treba da dobiju pristup bazi podataka. Takođe pratiti aktivnosti završene tokom tog vremenskog okvira i sprečavanje admistratora da dele lozinke. Iako administratori mogu smatrati da je deljenje lozinki zgodno, to čini efektivnu odgovornost i bezbednost baze podataka gotovo nemogućim. Pored toga preporučuju se sledeće mere bezbednosti: jake lozinke se moraju primeniti, hešovi lozinki moraju biti posoljenji i uskladišteni šifrovani, nalozi moraju biti zaključani nakon više pokušaja prijave, nalozi se moraju redovno pregledavati i deaktivirati ako osblje prelazi na različite uloge, napuštaju kompaniju ili više se ne zahteva isti nivo pristupa.

- **Testiranje bezbednosti baze podataka**

Kada se postavi bezbednosna infrastruktura baze podataka, mora se testirati u odnosu na stvarnu pretnju. Revizija ili izvođenje testova penetracije prema vašoj sopstenoj bazi podataka će razviti način razmišljanja sajber kriminalaca.

- **Koristiti praćenje baze podataka u realnom vremenu**

Kontinualno skeniranje baze podataka za pokušaje provale povećava sigurnost i omogućava brzo reagovanje na moguće napade. Konkretno, nadgledanje integriteta datoteka može pomoći da se evidentiraju sve radnje izvršene na serveru baze podataka i da se dobiju upozorenja na potencijalne povrede.

- **Koristiti zaštitne zidove za veb aplikacije i baze podataka**

Trebalo bi koristiti zaštitni zid da bi se zaštitio server baze podataka od bezbednosnih pretnji baze podataka. Takođe treba da se spreči da baza podataka pokreće izlazne veze osim ako ne postoji poseban razlog za to. Pored zaštite baze podataka zaštitnim zidom, mora se primeniti zaštitni zid veb aplikacije. To je zato što se napadi usmereni na veb aplikacije, uključujući SQL injekciju, mogu koristiti za dobijanje nezakonitog pristupa bazama podataka. Zaštitni zid baze podataka neće zaustaviti većinu napada na veb aplikacije, jer tradicionalni zaštitni zidovi rade na sloju mreže, dok slojevi veb aplikacija rade na sloju aplikacije (soj 7 OSI modela). [3]

2.5. Kontrole i politike

Pored implementacije slojevitih bezbednosnih kontrola u celom mrežnom okruženju, bezbednost baze podataka zahteva da se uspostave ispravne kontrole i smernice za pristup samoj bazi podataka. To uključuje:

- Administrativne kontrole za upravljanje instalacijom, promenama i upravljanjem konfiguracijom za bazu podataka.
- Preventivne kontrole za upravljanje pristupom, šifrovanjem, tokenizacijom i maskiranjem.
- Detektivske kontrole za praćenje aktivnosti baze podataka i alati za sprečavanje gubitka podataka. Ova rešenja omogućavaju identifikaciju i upozoravanje na anomalne ili sumnjive aktivnosti.

Politike bezbednosti baze podataka treba da budu integrisane i podržavaju opšte poslovne ciljeve, kao što su zaštita kritične intelektualne svojine i politike sajber bezbednosti i politike bezbednosti u oblaku. Bezbednosne kontrole, programi obuke i edukacije o svesti o bezbednosti, kao i strategije za testiranje penetracije i procene ranjivosti treba da budu uspostavljene kao podrška formalnim bezbednosnim politikama. [4]

3. SIGURNOST CASSANDRA BAZE PODATAKA

Cassandra pruža ove bezbednosne funkcije zajednici otvorenog koda:

- Autentifikacija zasnovana na interno kontrolisanom imenu uloge/lozinki
- Autorizacija zasnovana na upravljanju dozvolama za objekte
- Autentifikacija i autorizacija na osnovu JMKS korisničkog imena/lozinke
- SSL enkripcija
- Opšte mere bezbednosti.

U narednim poglavljima biće detaljno opisane svaka od navedenih funkcija.

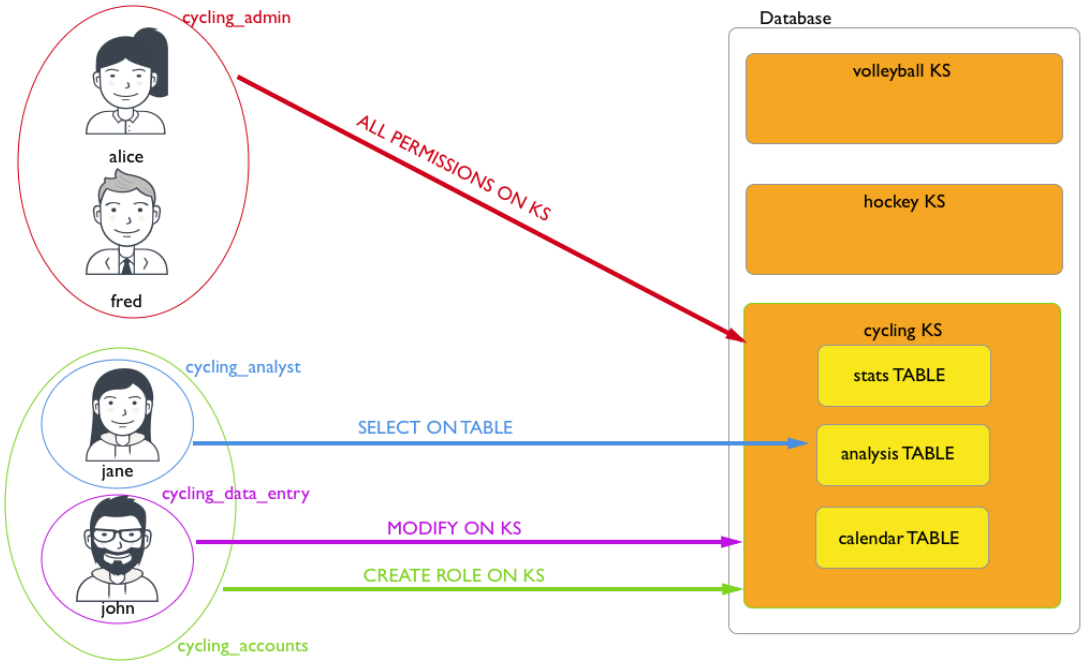
3.1. Autentifikacija zasnovana na interno kontrolisanom imenu uloge/imena

Cassandra autentifikacija je zasnovana na ulogama i interno se čuva u sistemskim tabelama Cassandre. Administratori mogu da kreiraju, menjaju, ispuštaju ili listaju uloge koristeći CQL komande, sa pridruženom lozinkom. Uloge se mogu kreirati sa privilegijama superkorisnika, nesuperkorisnika i privilegijama za prijavu. Interna autentikacija se koristi za pristup Cassandra

prostorima ključeva i tabelama, a cqlsh i DevCenter za autentifikaciju veza sa Cassandra klasterima i sstableloader-om za učitavanje SSTable-ova.

3.1.1. O internoj autentifikaciji

Kao i mnoge baze podataka, Cassandra koristi imena uloga i lozinke za internu autentifikaciju. Potvrda identiteta zasnovana na ulozi obuhvata i korisnike i ulog da bi se autorizaciji doneo niz korisnih funkcija. Uloge mogu predstavljati ili stvarne pojedinačne korisnike ili uloge koje ti korisnici imaju u administrisiranju i pristupu Cassandra klasteru. Na slici 2 se vidi primer uloga dodeljenih pojedincima i funkcijama:



Slika 2: Uloge dodeljene pojedincima i funkcijama [5]

Na primer, korisnik po imenu Alice je kreiran i dobio je privilegije za prijavu:

```
CREATE ROLE alice WITH PASSWORD = 'enjoyLife' AND LOGIN = true;
```

Treba znati da je korsnik kreiran kao uloga i da se ovaj korisnik može prijaviti u bazu podataka sa dodeljenim akreditivima za lozinku. Uloge se mogu kreirati sa privilegijama superkorisnika, nesuperkorisnika i privilegijama za prijavu. Privilegije superkorisnika dozvoljavaju ulozi da obavlja bilo koje operacije baze podataka. Primer kreiranja uloge kojoj će biti dat pristup funkcijama određenog ključnog prostora:

```
CREATE ROLE cycling_admin WITH PASSWORD = '1234abcd';

GRANT ALL PERMISSIONS ON KEYSPACE cycling TO cycling_admin;
```

Ova uloga, kada se dodeli korisniku, obezbediće određene privilegije korisniku na osnovu privilegija uloge; uloga kojoj je dodeljena ova uloga će naslediti privilegije cycling_admin. Ulozi cycling_admin se dodeljuju sve dozvole za cikliranje prostora ključeva u drugoj komandnoj liniji.

Kada je Alice dodeljena uloga cycling_admin, alice sada dobija sve dozvole za cikliranje prostora ključeva:

```
GRANT cycling_admin TO alice;
```

Pojedinačnom korisniku može biti dodeljen bilo koji broj uloga, kao što se svakoj funkcionalnoj ulozi mogu dodeliti dozvole druge uloge. U ovom primeru, uloga cycling_analyst ima mogućnost izbora podataka, a zatim dobija mogućnost izbora podataka u drugom hockey-u za stolom koda se dodeli uloga hockey_analyst.

```
CREATE ROLE cycling_analyst WITH PASSWORD = 'zyxw9876';

GRANT SELECT ON TABLE cycling.analysis TO cycling_analyst;

CREATE ROLE hockey_analyst WITH PASSWORD = 'Iget2seeAll';

GRANT SELECT ON TABLE hockey.analysis TO hockey_analyst;

GRANT hockey_analyst TO cycling_analyst;

GRANT cyclist_analyst TO jane;
```

Ako korisniku tada bude dodeljena uloga cycling_analyst, tada korisnik će moći da izabere podatke u dodatnom hockey na stolu. Gornja ilustracija bi bila modifikovana da pokaže da korisnik Jane sada ima pristup dva stola. Dozvole i SUPERUSER status su nasleđeni, ali LOGIN privilegija nije.

Važna promena koju takođe uvodi kontrola pristupa zasnovana na ulogama je da SUPERUSER je uklonjena potreba za privilegijama za obavljanje operacija upravljanja kornsikom/ulogom. Uloga može biti ovlašćena da dodeljuje i opoziva dozvole:

```
// Give cycling_accounts the right to create roles

GRANT CREATE ON ALL ROLES TO cycling_accounts;

// Give cycling_accounts the right to grant or revoke
permissions

GRANT AUTHORIZE ON KEYSPACE cycling TO cycling_accounts;

GRANT cyclist_accounts TO jane;

GRANT cyclist_accounts TO john;
```

Informacije o internoj autentifikaciji i autorizaciji se čuvaju u sledećim Cassandra tabelama (tabela 1):

Tabela 1: Cassandra tabele za internu autentifikaciju i autorizaciju

Tabela	Opis
system_auth.roles	Tabela u kojoj se čuva ime uloge, da li se uloga može koristiri za prijavljivanje, da li je uloga superkorisnik, koje druge uloge može biti član uloge i heš lozinku

	za ulogu.
system_auth.role_members	Tabela koja čuva uloge i članove uloge.
system_auth.role_permissions	Tabela koja čuva ulogu, resurs (ključni prostor, tabela) i dozvolu koju uloga ima za pristup resursu.
system_auth.recqlshSourceole_permissions_index	Tabela koja čuva ulogu i resurs za koji uloga ima predstavljenu dozvolu.

Cassandra je podrazumevano konfigurisana sa podrazumevanom ulogom superkorisnika i parom lozinke cassandra/cassandra. Koristeći ovu ulogu, dodatne uloge se mogu kreirati pomoću CQL komandi. Da bi se obezbedio sistem, ovu podrazumevanu ulogu je neophodno izbrisati kada se kreira superkorisnik koji nije podrazumevani. Kada su uloge i lozinke postavljene, Cassandra se može konfigurisati da koristi autentifikacuju u datoteci cassandra.yaml

Ako uloge postoje i Cassandra je konfigurisana da koristi autentifikaciju, Cassandra alati moraju da se izvrše sa opcionim opcijama autentifikacije: cqlsh sa autentifikacijom, DevCenter korišćenje SSL veza i DataStak drajveri, koji su proizvedeni od strane DataStak-a za rad sa Cassandrom. [5]

3.1.2. Konfigurisanje autentifikacije

Autentifikacija se može priključiti na Cassandri i konfiguriše se korišćenjem *authenticator* podešavanja u *cassandra.yaml*. Podrazumevano, Cassandra je konfigurisana tako *AllowAllAuthenticator* da ne vrši proveru autentifikacije i stoga ne zahteva nikakve akreditive. Koristi se za potpuno onemogućavanje autentifikacije. Treba imati na umu da je autentifikacija neophodan uslov Cassandrinog podsistema dozvola, tako da ako je autentifikacija onemogućena, zapravo su i dozvole. [6]

Podrazumevana distribucija takođe uključuje PasswordAuthenticator, koji čuva šifrovane akreditive u sistemskoj tabeli. Ovo se može koristiti za omogućavanje jednostavne provere autentičnosti korisničkog imena/lozinke.

Pre nego što omogućite autentifikaciju klijenta na klasteru, klijentske aplikacije treba da budu unapred konfigurisane sa njihovim predviđenim akreditivima. Kada se veza pokrene, server će tražiti akreditive samo kada je autentifikacija omogućena, tako da je podešavanje konfiguracije na strani klijenta unapred bezbedno. Nasuprot tome, čim server omogući autentifikaciju, svaki pokušaj povezivanja bez odgovarajućih akreditiva će biti odbijen, što može uzrokovati probleme dostupnosti za klijentske aplikacije. Kada su klijenti podešeni i spremni za omogućavanje autentifikacije, pratite ovu proceduru da biste je omogućili na klasteru. [6]

Koraci za konfiguranje autentifikacije su sledeći:

- Promena opcije autentifikatora u datotetci cassandra.yaml PasswordAuthenticatory:

  ```
  authenticator: PasswordAuthenticator
  ```

  Podrazumevano, opcija autentifikatora je podešena na:
  `AllowAllAuthenticator`
- Ponovno pokretanje Cassandre.

- Podrazumevano korišćenje ime superkorisnika i lozinke:

```
cqlsh -u cassandra -p cassandra
```

- Da bi bilo sigurno da je prostor ključeva uvek dostupan, potrebno je povećati faktor replikacije za prostor ključeva system_auth na 3 do 5 čvorova po centru podataka (preporučeno):

```
cqlsh> ALTER KEYSPACE "system_auth"

WITH REPLICATION = {'class' : 'NetworkTopologyStrategy',
'dc1' : 3, 'dc2' : 2};
```

Prostor system_auth ključeva koristi nivo konzistentnosti QUORUM kada proverava autentifikaciju za podrazumevanog korisnika cassandra. Za sve ostale kreirane korisnike, superkorisnike ili na drugi način, LOCAL_ONE nivo doslednosti se koristi za autentifikaciju. Imena centara podataka se razlikuju velika i mala slova. Ostavljanje podrazumevanog faktora ključeva system_auth dovodi do odbijanja pristupa klasteru ako se pojedinačna replika prostora ključeva pokvari. Za više centara podataka je obavezno postaviti klasu replikacije na *NetworkTopologyStrategy*.

- Nakon što je faktor replikacije prostora ključeva povećan, pokreće se nodetool repair da bi bili sigurni da se promena širi:

```
$ nodetool repair system_auth
```

- Ponovno pokretanje Cassandre.
- Korišćenje superkorisničko ime i lozinka:

```
cqlsh -u cassandra -p Cassandra
```

- Da bi se sprečilo narušavanje bezbednosti, zamena podrazumevanog superkorisnika, cassandra, drugim superkorisnikom sa drugim imenom:

```
cqlsh> CREATE ROLE <new_super_user> WITH PASSWORD =
'<some_secure_password>'

    AND SUPERUSER = true

    AND LOGIN = true;
```

Podrazumevani korisnik podrazumevano cassandra čita sa nivoom doslednosti QUOROM, dok drugi superkorisnik čita sa nivoom doslednosti LOCAL_ONE.

- Prijava kao novi superkorisnik:

```
cqlsh -u <new_super_user> -p <some_secure_password>
```

- Cassandra superkosirsnik se ne može izbrisati iz Cassandre. Da bi se neutralisao nalog, potrebno je promeniti lozinku u nešto dugačko i nerazumljivo i promeniti status u NOSUPERUSER:

```
cqlsh> ALTER ROLE cassandra WITH
PASSWORD='SomeNonsenseThatNoOneWillThinkOf'
    AND SUPERUSER=false;
```

- Kada se kreiraju neke nove uloge, spremno je ovlastiti te uloge za prisup objektima baze podataka.
- Preuzimanje autentifikacije uloge može biti skupa operacija. Ds bi se smanjilo opterećenje, može se prilagoiti period važenja za keširanje uloga pomoću opcije roles_validity_in_ms u datoteci cassandra.yaml (podrazumevano 2000ms):

```
roles_validity_in_ms: 2000
```

  Da bi se onemogućilo, postavlja se opcija na 0. Ova postavka se automarski deaktivira kada je autentifikator podešen na AllowAllAuthenticator.
- Konfigurisanje intervala osveženja na kešove uloga tako što će se postaviti opcija roles_update_interval_in_ms u datoteci cassandra.yaml (podrazumevano 2000ms):

```
roles_update_interval_in_ms: 2000
```

  Ako roles_validity_in_ms nije različit od nule, ovo podešavanje mora biti podešeno.

Sledeći koraci se primenjuju samo na Cassandru 3.4 i novije verzije:

- Preuzimanje autentifikacije akreditiva može biti skupa operacija. Da bi se smanjilo opterećenje, podešava se period važenja za keširanje akreditiva pomoću opcije credentials_validity_in_ms u datoteci cassandra.yaml (podrazumevano 2000ms):

```
credentials_validity_in_ms: 2000
```

  Da bi se onemogućilo, postavlja se ova opcija na 0. Ova postavka se automatski deaktivira kada je autentifikator podešen na AllowAllAuthenticator.
- Da bi se podesio interval osvežavanja za keš akreditiva, koristi se opcija credentials_update_interval_in_ms (podrazumevano 2000ms):

```
credentials_update_interval_in_ms: 2000
```

  Ako je credentials_validity_in_ms različit od nule, ovo podešavanje mora biti podešeno.
- Da bi se onemogućila konfiguracija kešova za autentifikaciju i autorizaciju (akreditive, uloge i dozvole) preko JMKS-a, potrebno je otkomentarisati sledeći red u jbm,options datoteci:

```
#-Dcassandra.disable_auth_caches_remote_configuration=true
```

  Nakon podešavanja ove opcije, opcije keša se mogu podesiti samo u datoteci cassandra.yaml. Da bi nova postavka stupila na snagu, ponovo je potrebno pokrenuti Cassandru. [5]

3.1.3. Korišćenje cqlsh sa autentifikacijom

Obično, nakon konfigurisanja autentifkacije, prijavljivanje na cqlsh zahteva opcije -u i -p za cqlsh komandu. Da bi se postavili akreditivi za upotrebu prilikom pokretanja cqlsh, kreira se ili menja datoteka .cassandra/cqlshrc. Kada je prisutna, ova datoteka prosleđuje podrazumevane informacije

za prijavu cqlsh. Datoteka cqlshrc.sample daje primer.

Procedura je sledeća:

- Kreiranje ili izmena datoteke cqlshrc koja navodi ime uloge i lozinku.

```
[authentication]
username = fred
password = !!bang!!$
```

- Čuvanje datoteke u kućnom direktorijumu / .cassandra i naziva se sa cqlshrc.
- Podešavanje dozvole za datoteku da bi se sprečio neolašćeni pristup, pošto se lozinka čuva u običnom tekstu. Datoteka mora biti čitljiva od strane korisnika koji pokreće cassandra.

```
$ chmod 440 home/.cassandra/cqlshrc
```

- Provera dozvole na početnoj stranici / .cassandra/ cqlshrc_history da bi bili sigurni da lozinke za običan tekst nisu ugrožene.

3.2. Autorizacija zasnovana na upravljanju dozvolama za objekte

Autorizacija daje privilegije pristupa operacijama Cassandra klastera na osnovu autentifikacije uloge. Autorizacija može dati dozvolu za pristup celoj bazi podataka ili ograničiti ulogu na pristup pojedinačnoj tabeli. Uloge mogu da daju ovlašćenje za autorizaciju drugih uloga. Uloge se mogu dodeliti ulogama. CQL komande GRANT i REVOKE se koriste za upravljanje autorizacijom.

Primer generi;ke sintakse za dodeljivanje dozvole korisnicima:

```
GRANT permission ON resource TO user
```

Postoje sledeće vrste dozvola koje se mogu dodeliti korisniku: ALL, ALTER, AUTHORIZIZE, CREATE, DROP, MODIFY, SELECT. Evo primera dodele dozvole korisniku:

```
Create user laura with password 'newhire';
grant all on dev.emp to laura;
revoke all on dev.emp to laura;
grant select on dev.emp to laura;
```

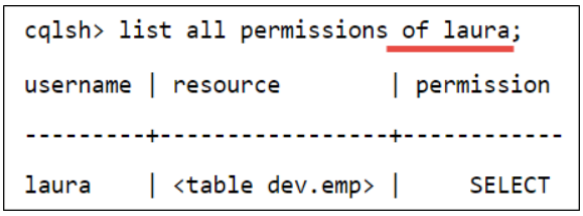
Novi korisnik ’laura’ je kreiran sa lozinkom ’newhire’. Primer gde korisnik ’laura’ pokušava da pristupi tabeli emp_bonus. Laura ima samo dozvolu da pristupi dev.emp i nema dozvolu za ovu tabelu dev.emp_bonus, zbog čega će biti vraćena greška (slika 3).

```
cqlsh:dev> select * from emp_bonus;
Bad Request: User laura has no SELECT permission on <table dev.emp_bonus> or any of its parents
```

Slika 3: Greška pri prustupu [7]

```
select* form emp_bonus;
```

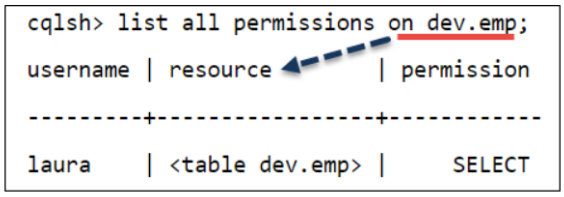
Ovim upitom se može dobiti lista svih dozvola koje su dodeljene korisniku. Evo primera dobijanja infomacija o dozvoli (slika 4):



Slika 4: Lista dozvola

```
list all permissions of laura;
```

Takođe se mogu navesti sve dozvole na resursu. Evo primera dobijanja dozvole (slika 5):



Slika 5: [7]

```
list all permissions on dev.emp;
```

3.2.1. Dozvole za objekte

Dozvole za objekte mogu biti dodeljene korišćenjem Cassandrinog internog mehanizma autorizacije za sledeće objekte: keyspace, table, funkcija, agregat, uloge i Mbeans (u Cassandra 3.6 i novijim verzijama). Ovlašćene uloge sa lozinkama uskladištenim u Cassandri su ovlašćeni selektivni pristup. Dozvole se čuvaju u Cassandrinim tabelama. Dozvola se može konfigurisati sa CQL komandama CREATE, ALTER, DROP, SELECT, MODIFY i DESCRIBE, koje se koriste za interakciju sa bazom podataka. Komanda EXECUTE se može koristiti za davanje dozvole ulozi za komande SELECT, INSERT i UPDATE. Pored toga, AUTHORIZE komanda se može koristiti za davanje dozvole za ulogu GRANT, REVOKE ili AUTHORIZE dozvole druge uloge.

Pristup za čitanje ovim sistemskim tabelama se implicitno daje svakom autentifikovanom korisniku ili ulozi jer te tabele koristi većina Cassandrinog alata:

- system_shema.keyspace
- system_shema.columns
- system_shema.tables
- system.local
- system.peers. [5]

3.2.2. Konfigurisanje interne autorizacije

CassandraAuthorizer je jedna od mogućih implementacija IAuthorizer. Njegova prednost je u tome što čuva dozvole u system_auth.permissions tabeli kako bi podržao sve CQL izjave vezane za autorizaciju. Da bi se aktivirao, menja se authorizer opcija u cassandra.yaml da bi se koristio CassandraAuthorizer.

Procedura je sledeća:

- U datoteci cassandra.yaml, komenarisati podrazumevanu vrednost AllowAllAuthorizer i dodavanja CassandraAuthorizer:
  ```
  authorizer: CassandraAuthorizer
  ```
  Može se koristit bilo koji autentifikatr osim AllowAll.
- Povećanje faktora replikacije za system_auth prostor ključeva ako već nije konfigurisan.
- Preuzimanje dozvola uloge može biti skupa operacija. Dozvole za uloge se mogu keširati da bi se smanjio teret. Podešavanje perioda važenja za keširanje dozvola tako što će se postaviti opcija permissions_validity_in_ms u datoteci cassandra.yaml. Podrazumevana vrednost je 2000ms. Keširanje se može onemogućiti postavljanjem opcije na 0. Ovo podešavanje je automatski onemogućeno ako je autorizator podešen na AllowAllAuthorizer
  ```
  permissions_validity_in_ms: 2000
  ```

- Interval osvežavanja za kešove uloga se takođe može konfigurisati postavljanjem opcije permissions_update_interval_in_ms u datoteci cassandra.yaml. Podrazumevana vrednost je ista vrednost kao i permissions_validity_in_ms podešavanje. Ako permissions_validity_in_ms je različit od nule, ovo podešavanje mora biti podešeno. [5]
  ```
  permissions_update_interval_in_ms: 2000
  ```

3.3. Autentifikacija i autorizacija na osnovu JMX korisničkog imena/lozinke

JMX (Java Managment Extensions) tehnologija pruža jenostavan i standardan način upravljanja i nadgledanja resursa koji se ondose na instancu Java virtuelne mašine (JVM). Ovo se postiže instrumentiranjem resursa sa Java objektima poznatim kao Managed Beans (MBeans) koji su registrovani na Mbean serveru. JMX autentifikacija čuva korisničko ime i povezane lozinke u dve datoteke, jednu za lozinke i jednu za pristup. JMX autentifikaciju koriste alati za nodetool i spoljni alati za nadgledanje kao što je jconsole. U Cassandri 3.6 i novijim verzijama, JMX autentikacija i autorizacija se mogu pronaći korišćenjem Cassandrinih internih mogućnosti autentifikacije i autorizacije. [5]

JMX autentifikacija i autorizacija omogućavaju selektivnim korisnicima pristup JMX alatima i JMX metrikama. U Cassandri 3.5 i ranijim verzijama, JMX je konfigurisan sa lozinkom i pristupnim datotekama. U Cassandri 3.6 i novijim verzijama, JMX veze mogu da koriste iste interne mehanizme autentifikacije i autorizacije kao CQL klijenti. Ako postoje korisnička imena i lozinke i Cassandra je konfigurisana da koristi autentifikaciju i autorizaciju, JMX alati moraju da se izvrše sa opcijama autentifikacije i autorizacije.

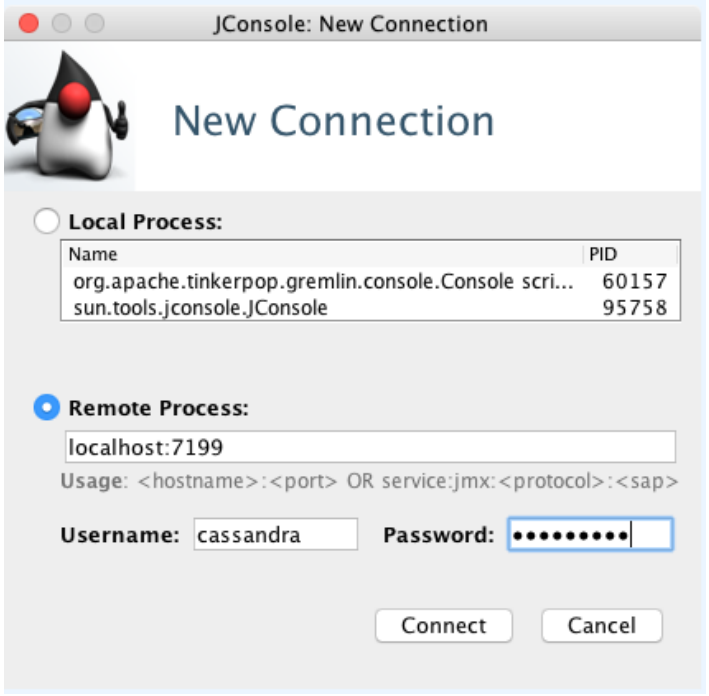
- Nodetool sa autentifikacijom
- Jconsole sa autentifikacijom.

Nakon konfigurisanja JMX autentifikacije, korišćenje nodetool zahteva opcije -u i -p za komande nodetool-a. Procedura je da se pokrene nodetool korišćenjem unapred konfigurisanog JMX korisničkog imena i lozinke:

```
$nodetool -u <username> -pw <password>
```

Za Cassandru 3.6 i novije verzije, korisničko ime i lozinka mogu biti interno konfigurisana Cassandra uloga i lozinka.

Kod korišćenja jconsole, nakon konfigurisanja JMX autentifikacije, jconsole zahteva korisničko ime i lozinku za dovršetak daljinske veze sa Cassandra klasterom. Koristi se odgovarajuća kombinacija imena/lozinke. Procedura je da se jconsole koristi unapred definisano ime i lozinku:



Slika 6: jconsole [5]

3.3.1. Omogućavanje JMX autentifikacije i autorizacije

Podrazumevano, JMX bezbednost je onemogućena i dostupna samo sa lokalnog hosta bez autentifikacije kao što je prikazano ispod iz datoteke cassandra-env.sh:

```
if [ "$LOCAL_JMX" = "yes" ]; then

  JVM_OPTS="$JVM_OPTS -Dcassandra.jmx.local.port=$JMX_PORT -
XX:+DisableExplicitGC"
```

Konfigurisanje JMX autentifikacije i autorizacije može se postići korišćenjem lokalne lozinke i pristupnih datoteka za postavljanje korisničkih imena, lozinki i dozvola za pristup. U Cassandri 3.6 i novijim verzijama, Cassandra interna autentifikacija i autarizacija mogu se opciono konfigurisati za JMX bezbednost. Ove dve metode rade za daljinsku autentifikaciju i autorizaciju, razlika je samo u lokaciji podešavanja konfiguracije u datoteci cassandra-env.sh. Lokalna konfiguracija se postavlja unutar `if ["$LOCAL_JMX" = "yes']; then` bloka u datoteci, dok se daljinska konfiguracija postavlja else blokom. [5]

3.4. SSL enkripcija

Cassandra obezbeđuje bezbednu komunikaciju između klijenata i klastera baze podataka, kao i između čvorova u klasteru. Omogućavaje SSL enkripcije osigurava da podaci u letu nisu ugroženi i da se bezbedno prenose. Šifrovanje od klijenata do čvora i od čvora do čvora se neizvesno konfiguriše. Cassandra alati se mogu konfigurisati da koriste SSL enkripciju. DataStek drajveri se mogu konfigurisati da obezbede saobraćaj između vozača i Cassandre.

Sloj bezbedne utičnice (SSL) je kriptografski protokol koji se koristi za obezbeđenje komunikacije između računara. Podaci se šifruju tokom komunikacije kako bi se sprečili slučajni ili namerni pokušaji čitanja podataka. SSL funkcioniše na sledeći način. Dva entiteta, softver ili harver, koji komuniciraju jedan sa drugim. Entiteti moraju da razmenjuju informacije da bi uspostavili poverenje između sebe. Svaki entitet koji će pružiti takve informacije mora imati generisani ključ koji se satoji od privatnog ključa koji samo entitet čuva i javnog ključa koji se može razmeniti sa drugim entitetima. Ako klijent želi da se poveže sa serverom, klijent zahteva bezbednu vezu i server šalje sertifikat koji uključuje njegov javni ključ. Ako klijent želi da se poveže sa serverom, klijent zahteva bezbednu vezu i server šalje sertifikat koji uključuje njegov javni ključ. Klijent proverava validnost sertifikata razmenom informacija sa serverom, koje server potvrđuje svojim privatnim ključem. Ako se želi dvosmerna validacija, ovaj proces se mora sprovesti u oba smera. Skladište ključeva i javni ključevi se čuvaju u skladištu poverenja. Za sisteme koji koriste autoritet sertifikata (CA), skladište poverenja može da skladišti sertifikate koje je potpisao CA radi verifikacije. I skladišta ključeva i skladišta poverenja imaju dodeljene lozinke, koje se nazivaju keypass i storepass. [5]

Apache Cassandra pruža ove funkcije SSL enkripcije:

- Šifrovana komunikacija od čvora do čvora – Šifrovanje od čvora do čvora se koristi za obezbeđenje podataka koji se prenose između čvorova u klasteru.
- Šifrovana komunikacija klijent-čvor – Šifrovanje od klijenata do čvora se koristi za obezbeđenje podataka prosleđenih između klijentskog programa, kao što je cqlsh, DevCenter ili nodetool, i čvorova u klasteru. [5]

Kako bi se omogućilo SSL od čvora do čvora, kao i od čvora do klijenta, potrebno je podesiti server_encription_options u datoteci cassanda.yaml.

Da bi se koristilo SSL šifrovanje za šifrovanje od klijenta do čvora ili šifrovanje od čvora do čvora, SSL sertifikati moraju da se generišu pomoću alata za ključeve. Ako se generišu sertifikati za jedan tip šifrovanja, nije potrebno ponovo ih generisati za drugi, za oba se koriste isti sertifikati. Svi čvorovi moraju imati sve relevantne SSL sertifikate na svim čvorovima. Skladište ključeva sadrži privatne ključeve. Skladište poverenja sadrži SSL sertifikate za svaki čvor. Sertifikati u truststore-u ne zahtevaju potpisivanje od strane pouzdanog i priznatog javnog autoriteta za sertifikaciju.

Procedura je sledeća:

- Generisanje par privatnih i javnih ključeva na svakom čvoru klastera. Koristiti pseudonim koji identifikuje čvor. Traži lozinku za skladištenje ključeva, dname (ime i prezuime, organizacinu jedinicu, grad, državu, zemlju) i lozinku za ključ. Dname treba da se generiše sa CN vrednošću kao IP adresom ili FQDN za čvor:

```
$keytool -genkey -keyalg RSA -alias node0 -validity 36500
-keystore keystore.node0
```

- Komanda za generisanje takođe može uključiti sve tražene informacije u komandnoj liniji. Ovaj primer koristi pseudonim od node0, ime skladišta ključeva od keystore.node0, koristi istu lozinku cassandra i za skladište ključeva i za ključ, i dname koje identifikuje IP adresu node0 kao 172.31.10.22.

```
$keytool -genkey -keyalg RSA -alias node0 -keystore
keystore.node0 -storepass cassandra -keypass cassandra -
dname "CN=172.31.10.22, OU=None, O=None, L=None, C=None"
```

- Izvoz javnog dela sertifikata u posebnu datoteku:

```
$keytool -export -alias cassandra -file node0.cer -
keystore .keystore
```

- Dodavanje node0.cer sertifikat u skladište poverenja node0 čvora koristeći keytool -import komandu:

```
$keytool -import -v -trustcacerts -alias node0 -file
node0.cer -keystore truststore.node0
```

- Cqlsh ne radi sa sertifikatom u generisanom formatu. Openssl se koristi za generisanje PEM datoteke sertifikata bez ključeva, node0.cer.pem i PEM datoteke ključa bez sertifikata, node0.key.pem. Prvo, skladište ključeva se uvozi u PQCS12 formatu u odredišno skladište ključeva, node0.p12 u primeru. Posle toga slede dve komande koje izdvajaju dve PEM datoteke

```
keytool -importkeystore -srckeystore keystore.node0 -
destkeystore node0.p12 -deststoretype PKCS12 -srcstorepass
cassandra -deststorepass cassandra

openssl pkcs12 -in node0.p12 -nokeys -out node0.cer.pem -
passin pass:cassandra

openssl pkcs12 -in node0.p12 -nodes -nocerts -out
node0.key.pem -passin pass:cassandra
```

- Za šifrovanje od klijenta do udaljenog čvora ili šifrovanja od čvora do čvora, korisi se alatka za kopiranje kao scp da se kopira node0.cer datoteke za svaki čvor. Uvoz datoteke u skladište poverenja nakon kopiranja na svaki čvor. Primer uvozi sertifikat za čvor 0 u skladište poverenja za čvor 1

```
$keytool -import -v -trustcacerts -alias node0 -file
node0.cer -keystore truststore.node1
```

- Potrebno je uveriti se da je fajl skladišta ključeva čitljiv samo za Cassandra demon, a ne za bilo kog korisnika sistema.
- Provera da li sertifikati postoje u datotekama skladišta ključeva i skladišta poverenja koristeći keytool -list. Primer pokazuje proveru sertifikata node1 u datoteci skladišta ključeva i sertifikata node0 i node1 u datoteci skladišta poverenja.

```
$keytool -list -keystore keystore.node1
keytool -list -keystore truststore.node1
```

- Uvoz korisničkih sertifikata u skladište poverenja samog čvora koristteći keytool:

```
$keytool -import -v -trustcacerts -alias <username> -file
<certificate file> -keystore .truststore
```

Da bi se SSL sertifikati koristili za šifrovanje od klijenta do čvora ili šifrovanje od čvora do čvora, SSL sertifikati moraju da se generišu pomoću openssl ili keytool-a. Da bi se potvrdili sertifikati, može se generisati samopotpisani autoritet sertifikata (CA) za proizvodnu upotrebu sa Apache Cassandra. Sertifikati generisani korišćenjem ovih uputstava mogu se koristiti za enkripciju među čvorovima i za šifrovanje od klijenata do čvora. Za šifrovanje internood-a, svi čvorovi moraju imati skladište poverenja koje obezbeđuje lanac poverenja za CA. Sertifikate u skladištu poverenja može ili potpisati samopotpisani autoritet za izdavanje sertifikata koji se ovde koristi ili od poverenja i priznatog javnog autoriteta za sertifikate. [5]

3.5. Opšte mere bezbednosti

Tipično, proizvodni Cassandra klasteri će imati zatvorene sve nebitne portove zaštitnog zida. Neki portovi moraju biti otvoreni da bi čvorovi mogli da komuniciraju u klasteru. Ovi portovi su detaljni.

Sledeći portovi moraju biti otvoreni da bi se omogućila dvosmerna komunikacija između čvorova, uključujući određene Cassandra portove. U skladu sa tim konfiguriše se zaštitini zid koji radi na čvorovima u Cassandra klasteru. Bez otvorenih portova kao što je prikazano, čvorovi će delovati kao samostalni server baze podataka i neće se pridružiti Cassandra klasteru.

Tabela 2 : Javni port

Broj porta	Opis
22	SSH port

Tabela 3: Cassandra inter-node portovi

Broj porta	Opis
7000	Cassandra inter-node klister komunikacija
7001	Cassandra SSL komunikacija klastera među čvorovima
7199	Cassandra JMKS port za nadgledanje

Tabela 4: Cassandra klijent portovi

Broj porta	Opis
9042	Cassandra klijent port
9160	Cassandra klijent port
9142	Podrazumevano za native_transport_port_ssl, korisno kada us potrebne i šifrovane i nešifrovane veze

3.6. Keširanje

Omogućavanje autentifikacije i autorizacije dodatno opterećuje klaster čestim čitanjem iz system_auth tabela. Ova očitavanja su na kritičnim putanjama mnogih klijentskih operacija, i stoga imaju potencijal da ozbiljno utiču kvalitet usluge. Da bi se ovo ublažilo, podaci o autentifikaciji kao što su akreditivi, dozvole i detalji o ulozi se keširaju tokom perioda koji se može konfigurisati. Keširanje se može konfigurisati (pa čak i onemogućiti) sa cassandra.yaml ili pomoću JMX klijenta. JMX interfejs takođe podržava poništavanje različitih keš memorija, ali sve promene napravljene preko JMKS-a nisu trajdne i biće ponovo pročitane cassanda.yaml kada se čvor pokrene.

Svaki keš ima 3 opcije koje se mogu podesiti:

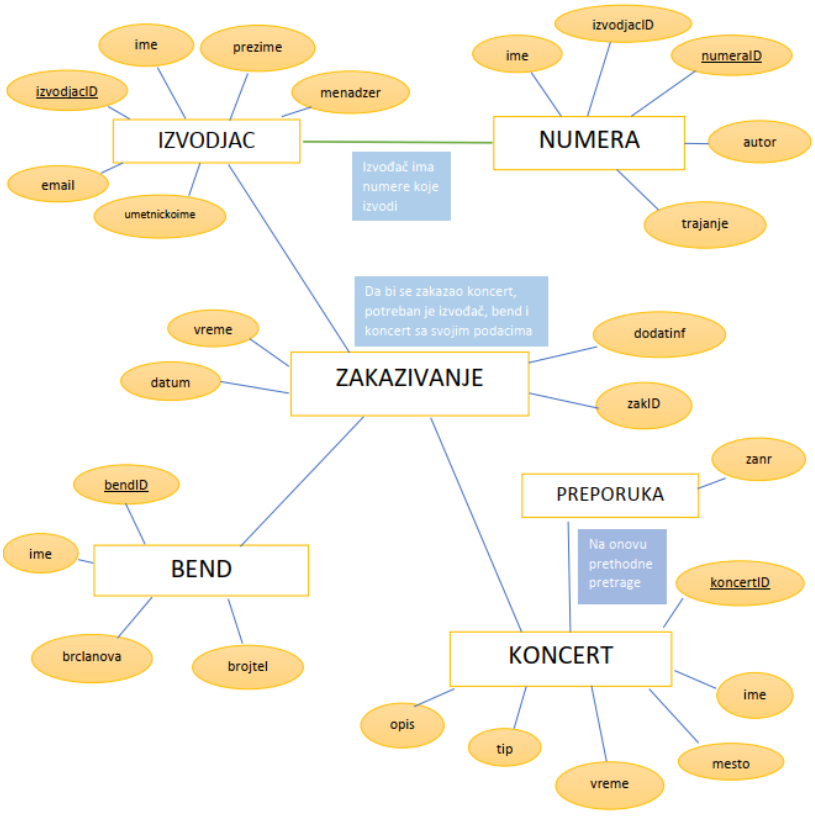
- Rok važenja – Kontroliše istek unosa u keš memoriju. Nakon ovog perioda, unosi se poništavaju i uklanjanju iz keša.
- Refresh Rate – Kontroliše brzinu kojom se pozadinsko čitanje obavlja da bi se pokupile sve promene osnovnih podataka. Dok se ova asinhronizovana osveženja obavljaju, keš memorija će nastaviti da služi zastarele podatke. Obično će ovo biti podešeno na kraće vreme od perioda važenja.
- Max Entries – Kontroliše gornju granicu veličine keša.

4. PRAKTIČNA PRIMENA

U ovom poglavlju nalazi se opis kreiranja baze podataka u Cassandri, dodavanja podataka, a nakon toga, detaljan opis primene mera sigurnosti nad Cassandra bazom podataka i opis postupka kojim se baza podataka osigurava od različitih neodgovornih ili namernih upada i zloupotrebe podataka.

4.1. Skup podataka baze podataka

Skup podataka odnosi se na zakazivanje online koncerata, gde se nalaze podaci o koncertima, izvođačima, bendovima, numerama koje izvođači izvode i zakazivanju. Na slici je prikazan dijagram koji prikazuje sve entitete sa atributima, gde su naznačeni atributi koji predstavljaju primarne ključeve.



Slika 7: Dijagram

- Prvi korak je kreiranje keyspace. To se čini na sledeći način:

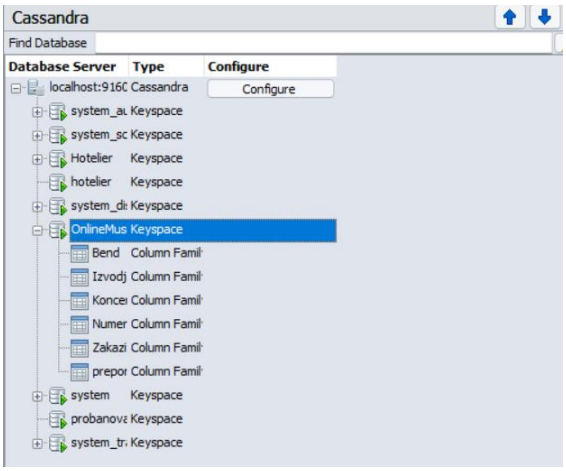
```
create keyspace OnlineMusicConcert;
```

- Kada je keyspace uspešno kreiran, mogu se kreirati column family unutar njega. Neophodno je kreirati Koncert, Izvodjac, Bend, Numera, Zakazi, preporuka. I to na sledeći način:

```
CREATE TABLE "Koncert" (
    "koncertID" text,
    ime text,
    opis text,
    organizator text,
    sponzor text,
    tip text,
    PRIMARY KEY ("koncertID")
)

CREATE TABLE "Zakazivanje" (
     "zakID" text,
     "izvodjacID" text,
     "bendID" text,
     "koncertID" text,
```

```
    vreme text,
    datum text,
    dodatinf text,
    PRIMARY KEY("zakID")
)
```

Na isti način kreiraju se i ostale tabele. Kada su sve uspešno kreiranje u okviru keyspace-a OnlineMusicConcert u NoSQL Viewer-u može se sa leve strane videti struktura kao na slici 8:

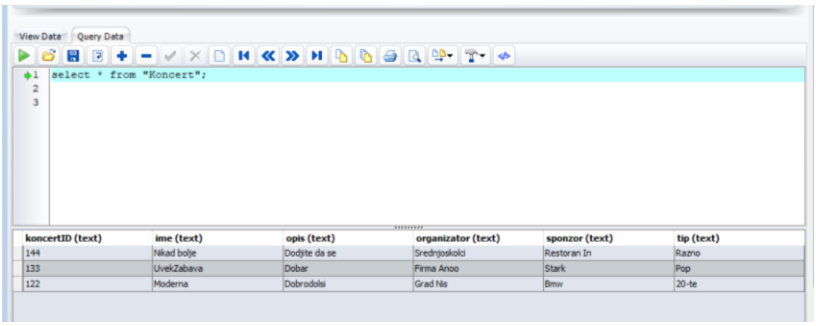


Slika 8: OnlineMusicConcert keyspace

- Kada su uspešno kreirane tabele, mogu se dodati podaci. Ispod se nalaze primeri nekih od njih:

```
insert into "Koncert" ("koncertID",ime,opis,organizator,sponzor,tip) values
('122','Moderna','Dobrodolsi','Grad Nis','Bmw','20-te');

insert into "Koncert" ("koncertID",ime,opis,organizator,sponzor,tip) values ('133','UvekZabava','Dobar
provod.','Firma Anoo','Stark','Pop');

insert into "Koncert" ("koncertID",ime,opis,organizator,sponzor,tip) values ('144','Nikad bolje','Dodjite da
se zabavimo','Srednjoskolci','Restoran In','Razno');
```

- Naredbom `select * from "Koncert";` želimo da prikažemo sve podatke koji su uneti. Na slici 9 se vidi rezultat ovog upita:



koncertID (text)	ime (text)	opis (text)	organizator (text)	sponzor (text)	tip (text)
144	Nikad bolje	Dodjite da se	Srednjoskolci	Restoran In	Razno
133	UvekZabava	Dobar	Firma Anoo	Stark	Pop
122	Moderna	Dobrodolsi	Grad Nis	Bmw	20-te

Slika 9: Rezultat upita

Na potpuno isti način vrši se dodavanje podataka u ostalim tabelama. Primeri takvih upita nalaze se u pratećem dokumentu, koji se dostavlja uz ovaj rad, pod nazivom 'Create and insert.docx'.

4.2. Mere sigursnosti nad bazom podataka

Izgled dela cassandra.yaml fajla koji se tiče autorizacije i autentifikacije prikazan je na slici 10. To su defaultne postavke koje se nalaze u ovom fajlu.

```
#
# - AllowAllAuthenticator performs no checks - set it to disable authentication.
# - PasswordAuthenticator relies on username/password pairs to authenticate
#   users. It keeps usernames and hashed passwords in system_auth.roles table.
#   Please increase system_auth keyspace replication factor if you use this authenticator.
#   If using PasswordAuthenticator, CassandraRoleManager must also be used (see below)
authenticator: AllowAllAuthenticator

# Authorization backend, implementing IAuthorizer; used to limit access/provide permissions
# Out of the box, Cassandra provides org.apache.cassandra.auth.{AllowAllAuthorizer,
# CassandraAuthorizer}.
#
# - AllowAllAuthorizer allows any action to any user - set it to disable authorization.
# - CassandraAuthorizer stores permissions in system_auth.role_permissions table. Please
#   increase system_auth keyspace replication factor if you use this authorizer.
authorizer: AllowAllAuthorizer

# Part of the Authentication & Authorization backend, implementing IRoleManager; used
# to maintain grants and memberships between roles.
```

Slika 10: cassandra.yaml fajl

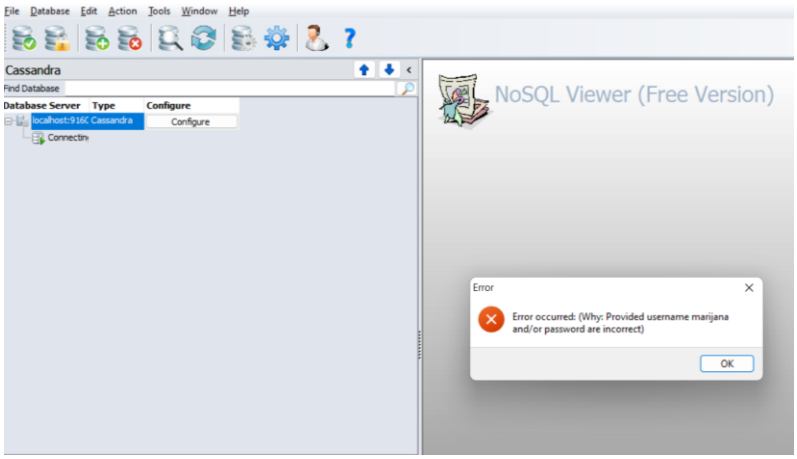
Kako bi se autentifikacija omogućila neophodno je promeniti deo gde je AllowAuthenticator sa PasswordAuthenticator (slika 11).

```
# - PasswordAuthenticator relies on username/password pairs to authenticate
#   users. It keeps usernames and hashed passwords in system_auth.roles table.
#   Please increase system_auth keyspace replication factor if you use this authenticator.
#   If using PasswordAuthenticator, CassandraRoleManager must also be used (see below)
authenticator: PasswordAuthenticator

# Authorization backend, implementing IAuthorizer; used to limit access/provide permissions
# Out of the box, Cassandra provides org.apache.cassandra.auth.{AllowAllAuthorizer,
# CassandraAuthorizer}.
#
# - AllowAllAuthorizer allows any action to any user - set it to disable authorization.
# - CassandraAuthorizer stores permissions in system_auth.role_permissions table. Please
#   increase system_auth keyspace replication factor if you use this authorizer.
authorizer: CassandraAuthorizer

# Part of the Authentication & Authorization backend, implementing IRoleManager; used
```

Slika 11: Konfiguracija cassandra.yaml fajla

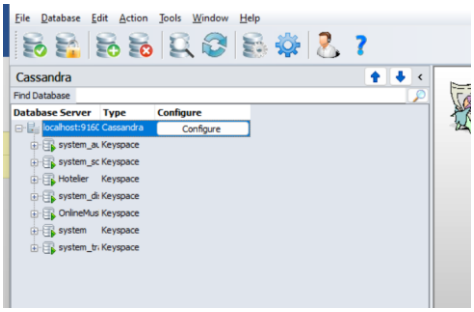
Podrazumevani korisnik i lozinka za pristup Cassandra bazi podataka su: cassandra i cassandra.

Kako koristimo NoSQLViewer kao alat za pristup Cassandra bazi podataka, pri pokušaju pristupa Cassandra bazi podataka sa pogrešnim imenom ili lozinkom, dobiće se sledeći izlaz (slika 12):



Slika 12: NoSQLViewer

Kao što se sa leve strane može videti, nemoguće je dobiti prikaz keyspace-ova Cassandra baze podataka, sa svim svojim tabelama i unetim podacima. Tek kada se uspešno izvrši prijava na bazu podataka sa tačnim korisničkim imenom i lozinkom, tek tada se mogu prikazati svi podaci (slika 13).



Slika 13: Uspešno logovanje

Ukoliko je potrebna izmena imena i lozinke korisnika, to se može učiniti na primer na sledeći način:

```
ALTER USER cassandra WITH PASSWORD 'cassandra123';
```

Pri narednoj prijavi, neophodno je uneti nove podatke za prijavu, kako bi se moglo pristupiti podacima.

Na primer, korisnik po imenu *marijana* je kreiran i dobio je privilegije za prijavu:

```
CREATE ROLE marijana WITH PASSWORD = 'marijanaa123' AND LOGIN =
true;
```

Treba znati da je korsnik kreiran kao uloga i da se ovaj korisnik može prijaviti u bazu podataka sa dodeljenim akreditivima za lozinku. Uloge se mogu kreirati sa privilegijama superkorisnika, nesuperkorisnika i privilegijama za prijavu. Privilegije superkorisnika dozvoljavaju ulozi da obavlja bilo koje operacije baze podataka. Primer kreiranja uloge kojoj će biti dat pristup funkcijama određenog ključnog prostora:

```
CREATE ROLE base_admin WITH PASSWORD = '1234abcd';
```

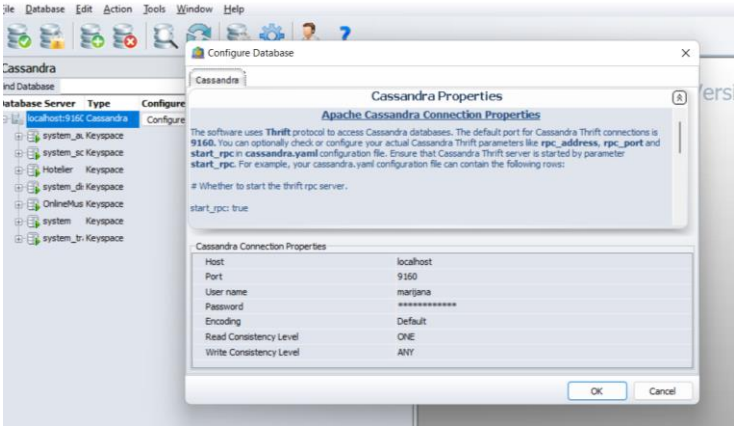
```
GRANT ALL PERMISSIONS ON KEYSPACE "OnlineMusicConcert" TO
base_admin;
```

Ova uloga, kada se dodeli korisniku, obezbediće određene privilegije korisniku na osnovu privilegija uloge; uloga kojoj je dodeljena ova uloga će naslediti privilegije base_admin. Ulozi base_admin se dodeljuju sve dozvole za cikliranje prostora ključeva u drugoj komandnoj liniji.

Kada je Marijana dodeljena uloga base_admin, *marijana* sada dobija sve dozvole za cikliranje prostora ključeva:

```
GRANT base_admin TO marijana;
```

Kada se ulogujemo kao korisnik *marijana*, videćemo da taj korisnik ima pristup svim tabelama keyspace-a OnlineMusicConcert, što se vidi i na slici 14:



Slika 14: Ulogovan korisnik

Pojedinačnom korisniku može biti dodeljen bilo koji broj uloga, kao što se svakoj funkcionalnoj ulozi mogu dodeliti dozvole druge uloge. U ovom primeru, uloga base_analyst ima mogućnost izbora podataka, a zatim dobija mogućnost izbora podataka u drugom hockey-u za stolom koda se dodeli uloga concert_analyst.

```
CREATE ROLE base_analyst WITH PASSWORD = 'zyxw9876';

****GRANT SELECT ON TABLE OnlineMusicConcert.koncert TO
base_analyst;

CREATE ROLE second_analyst WITH PASSWORD = 'Iget2seeAll';

****GRANT SELECT ON TABLE OnlineMusicConcert.koncert TO
second_analyst;

GRANT second_analyst TO base_analyst;

CREATE ROLE jovan WITH PASSWORD = 'jovan123' AND LOGIN = true;

GRANT base_analyst TO jovan;
```

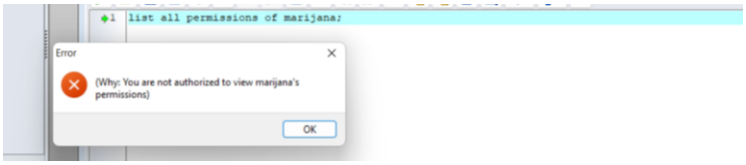
Ako korisniku tada bude dodeljena uloga base_analyst, tada korisnik će moći da izabere podatke

u dodatnom hockey na stolu. Gornja ilustracija bi bila modifikovana da pokaže da korisnik Jovan sada ima pristup dva stola. Dozvole i SUPERUSER status su nasleđeni, ali LOGIN privilegija nije.

Važna promena koju takođe uvodi kontrola pristupa zasnovana na ulogama je da SUPERUSER je uklonjena potreba za privilegijama za obavljanje operacija upravljanja kornsikom/ulogom. Uloga može biti ovlašćena da dodeljuje i opoziva dozvole:

```
// Give base_accounts the right to create roles

CREATE ROLE base_accounts WITH PASSWORD = 'baseacc';

GRANT CREATE ON ALL ROLES TO base_accounts;

// Give base_accounts the right to grant or revoke permissions

GRANT AUTHORIZE ON KEYSPACE “OnlineMusicConcert” TO
base_accounts;

CREATE ROLE maria WITH PASSWORD = 'mariamaria' AND LOGIN = true;

CREATE ROLE julian WITH PASSWORD = 'julian56' AND LOGIN = true;

GRANT base_accounts TO maria;

GRANT base_accounts TO julian;
```

Sada, kada smo prijavljeni kod korisnika 'maria' i želimo da pregledamo dozvole korisnika ’marijana’, videćmo da je to nemoguće, jer korisnik maria nema privilegije da pregleda dozvole drugih korisnika i rezultat će biti kao na slici 15:



Slika 15: Pregled dozvola korisnika

Kada se ovaj upit izvrši kod korisnika ’cassandra’, biće uspešan iz razloga što je to superkorisnik.

5. ZAKLJUČAK

Bezbednost baze podataka je složen i izazovan poduhvat koji uključuje sve aspekte tehnologija i praksi informacione bezbednosti.Takođe je prirodno u suprotnosti sa uportrebljivošću baze podataka. Što je baza podataka pristupačnija i upotrebljivija, to je ranjiva na bezbednosne pretnje, što je baza podataka neranjivija na pretnje, teže joj je pristupiti i koristiti.

Tema ovog rada bila je detaljan opis problema sa kojima je moguće suočiti se svaka od baza podataka, pretnji za njene podatke, koji su pod rizikom od krađe stalno. Postoje mnogi mehanizmi zaštite podataka od neželjenih upada, namernih ili slučajnih. Stoga, svaka od baza podataka primenjuje različite mehanizme zaštite svojih podataka, postoje posebne kontrole i politike koje se primenjuju.

Cassandra baza podataka poseduje nekoliko funkcija za zaštitu svojih podataka, a to su: autorizacija, autentifikacija, SSL enkripcija i opšte mere bezbednosti. Detaljno su opisane ove funkcije i na koji način se vrši njihova konfiguracija radi sigurne zaštite podataka.

U četvrtom poglavlju opisana je baza podataka koja se odnosi na zakazivanje koncerata, gde su entiteti koncert, mesto održavanja, bend, numera koju bend izvodi,.. Dat je prikaz kreiranja različith korisnika, sa različitim pravom pristupa podacima, gde se postiže očuvanje podataka i dozvola različitim korisnicima/zaposlenima u organizaciji pristup samo dozvoljenim podacima.

Cassandra baza podataka poseduje mehanizme za zaštitu svojih podataka. Novije verziju uvode još neke dodatne funkcije. Kako napredak tehnologija i svakodnevni rizik od krađe ili zloupotrebe podataka raste, sigurno će biti potrebe za uvođenjem dodatnih mehanizama i funkcija za zaštitu poverljivih podataka.

6. LITERATURA

[1] Database Security, dostupno na: https://www.techopedia.com/definition/29841/database-security

[2] Dumančić Robert, Sigurnost baze podataka, Osijek 2019, https://zir.nsk.hr/islandora/object/etfos%3A2274/datastream/PDF/view

[3] Database Security, dostupno na: https://www.imperva.com/learn/data-security/database-security/

[4] Database Security, dostupno na: https://www.ibm.com/cloud/learn/database-security

[5] Security, dostupno na: https://docs.datastax.com/en/cassandra-oss/3.0/cassandra/configuration/secureTOC.html

[6] Security, dostupno na: https://cassandra.apache.org/doc/trunk/cassandra/operating/security.html?fbclid=IwAR0Un2gnMYKlV769sGq6sXKBHrUIRPpAAcjMSUSmN-Sn4uA_d1CY9FbKF_o

[7] Cassandra JMX Authentication & Authorization: Create User, dostupno na: https://www.guru99.com/cassandra-security.html