ŠIAULIŲ UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
INFORMATIKOS KATEDRA

Žilvinas Brobliauskas
Informatikos specialybės magistro studijų II kurso dieninio skyriaus studentas

RELIACINIŲ DUOMENŲ BAZIŲ SAUGUMO MODELIO TYRIMAS

THE RESEARCH ON SECURITY MODEL OF RELATIONAL DATABASES

MAGISTRO DARBAS

Darbo vadovas:
doc. V. Sirius

Recenzentas:
lekt. V.Giedrimas

Šiauliai, 2009

Tvirtinu, jog darbe pateikta medžiaga nėra plagijuota ir paruošta naudojant literatūros sąraše pateiktus informacinius šaltinius bei savus tyrimų duomenis.

Žilvinas Brobliauskas

Turinys

Įvadas

Didėjant kompiuterių spartai atsiranda galimybė apdoroti vis didesnį duomenų kiekį, tačiau dirbant su milžiniškomis duomenų apimtimis iškyla problema kaip šiuos duomenis saugoti ir tvarkyti. Ši problema dažniausiai išsprendžiama pasitelkus reliacines duomenų bazes, kuriose duomenys saugomi naudojant reliacinius ryšius.

Šių dienų pasaulyje informacija yra vertinga ir reiškia labai daug, dėl to nenuostabu, kad vis dažniau ji tampa įvairių nusikaltėlių taikiniu, todėl informacijos apsauga įgauna vis didesnę reikšmę. Dėl savo realizacijos paprastumo dažniausiai yra taikomas klasikinis vienašalio saugumo modelis, belaikąs duomenų bazių sistemą ir jos vartotojus patikimais duomenų bazės atžvilgiu ir saugantis juos nuo kenkėjų išorėje, tačiau gali pasitaikyti atvejų, kuomet duomenis reikia saugoti ne tik nuo išorės kenkėjų, bet ir nuo pačios duomenų bazių sistemos; t.y. kartais būtina leisti duomenų valdymo sistemai atlikti duomenų tvarkymą nepaliekant jai galimybės interpretuoti pačių duomenų – tokiu atveju tenka taikyti daugiašalio saugumo modelį.

Siekiant užtikrinti duomenų slaptumą, kai jie saugomi nesaugioje vietoje, dažniausiai į pagalbą pasitelkiama kriptografija. Deja duomenų šifravimo taikymas daugiašalio saugumo modelyje nėra paprastas, kadangi duomenų bazių valdymo sistema neturėdama galimybės iššifruoti duomenis privalo atlikti paiešką ir skaičiavimų pagal vartotojų užklausas. Šiame darbe yra atliekamas teorinis daugiašalio reliacinių duomenų bazių saugumo modelio, pagrįsto duomenų šifravimu, tyrimas.

Darbo tikslas: reliacinių duomenų bazių saugumas.

Darbo uždaviniai:

1. Suformuluoti daugiašalio reliacinių duomenų bazių saugumo modeliui keliamus reikalavimus.
2. Pasiūlyti daugiašalio reliacinių duomenų bazių saugumo modelį, tinkantį plačiausiai naudojamoms reliacinėms duomenų bazių valdymo sistemoms.
3. Nustatyti pasiūlyto modelio privalumus ir trūkumus.
4. Sukurti pasiūlyto modelio realizaciją, įrodančią modelio veikimą.

I. Temos analizė

1. Reliacinis duomenų modelis ir duomenų bazės

Duomenų bazė – pagal tam tikrą schemą (duomenų modelį) struktūrizuotas duomenų rinkinys.

[1 psl. 1; 2 psl. 2]

Plačiausiai žinomi šie duomenų modeliai:

1. *Plokščiasis (lentelės) modelis*: duomenys saugomi vienoje dvimatėje lentelėje;
2. *Hierarchinis modelis*: duomenys saugomi medžio struktūroje;
3. *Tinklinis modelis*: duomenys saugomi tinklo struktūroje;
4. *Esybių sąryšių modelis (ERM)*: duomenys išskirstomi į esybes, jų atributus ir sąryšius tarp atributų.
5. *Objektinis modelis*: duomenys saugomi kaip objektai;
6. *Reliacinis modelis*: duomenys saugomi reliacinėse lentelėse remiantis predikatų logika ir aibių teorija.

[1 psl. 7-11]

Reliacinė duomenų bazė – pagal reliacinį duomenų modelį struktūrizuotas duomenų rinkinys.

Reliacinį duomenų modelį pirmasis formaliai savo darbuose [3][4] aprašė E.F.Codd. Šis modelis pagrįstas prielaida, kad visi duomenys gali būti išreikšti kaip n-nariai sąryšiai, kurie yra n duomenų sričių Dekarto sandaugos poaibiai.

Sąryšis (dar vadinamas reliacine lentele) – pagrindinis reliacinio modelio elementas, sudarytas iš atributų antraštės ir aibės n-arių gretinių su atributų reikšmėmis. Atributų antraštė – gretinys iš n porų, kurias sudaro atributo ir atributo tipo pavadinimai.

[1 psl. 79-81; 5 psl. 1-5]

Reliacinėje lentelėje n-ariai (atributų reikšmių) gretiniai priklauso aibei, dėl to pagal aibės apibrėžimą joje negali egzistuoti du vienodi n-ariai gretiniai. Gretinių unikalumas užtikrinamas aprašant kandidatinius raktus. *Kandidatinis raktas* – tokia reliacinės lentelės atributų aibė, tenkinanti sąlygą (bet neturinti tikrojo poaibio tenkinančio tą pačią sąlygą), jog tų atributų reikšmių gretinys bus unikalus visoje reliacinėje lentelėje. Kiekviena reliacinė lentelė gali turėti nuo 1 iki keleto kandidatinių raktų. Kandidatinis raktas, išskirtinai nurodytas, kaip priemonė identifikuoti skirtingus atributų reikšmių gretinius, yra vadinamas *pirminiu raktu*.

[1 psl. 85-87; 5 psl. 22-26]

Reliaciniame modelyje ryšiai tarp atskirų lentelių išreiškiami ne kokiomis nors specialiomis jungtimis, o vienodomis lentelių atributų reikšmėmis. *Svetimasis raktas* – reliacinės lentelės atributų aibė, atitinkanti kitos reliacinės lentelės kandidatinį raktą.

[1 psl. 87; 5 psl. 22-26]

Duomenimis pagal reliacinį modelį yra manipuliuojama pasitelkus *reliacinę algebrą*. Pagrindiniai jos operatoriai:

1. Primityvieji operatoriai:
    a. Aibių operatoriai:
        i. *Sąjungos operatorius* grąžina reliacinę lentelę, kurios n-arių gretinių aibė lygi kitų dviejų reliacinių lentelių, turinčių tą pačią antraštę, n-arių gretinių aibių sąjungai.
        ii. *Skirtumo operatorius* grąžina reliacinę lentelę, kurios n-arių gretinių aibė yra dviejų kitų reliacinių lentelių, turinčių tas pačią antraštę, n-arių gretinių aibių skirtumas.
        iii. *Sankirtos operatorius* grąžina reliacinę lentelę, kurios n-arių gretinių aibė yra dviejų kitų reliacinių lentelių, turinčių tas pačią antraštę, n-arių gretinių aibių sankirta.
        iv. *Dekarto sandaugos operatorius* grąžina reliacinę lentelę, kurios n-arių gretinių aibė yra dviejų kitų reliacinių lentelių n-arių aibių Dekarto sandauga.
    b. *Projekcijos* operatorius grąžina reliacinę lentelę, gautą iš kitos reliacinės lentelės paliekant tik nurodytus atributus.
    c. *Selekcijos operatorius* grąžina reliacinę lentelę, gautą iš kitos reliacinės pašalinus n-arius gretinius, kuriems priklauso atributų reikšmės, neatitinkančios tam tikras nurodytas sąlygas.
    d. *Pervadinimo operatorius* grąžina reliacinę lentelę, kuri nuo lentelės, kuriai buvo pritaikytas pervadinimo operatorius, tesiskiria kažkurio atributo vardu.
    e. Agregatiniai operatoriai:
        i. *Minimumo operatorius* grąžina reliacinę lentelę, turinčią 1-arį su nurodyto atributo mažiausia reikšme.
        ii. *Maksimumo operatorius* grąžina reliacinę lentelę, turinčią 1-arį su nurodyto atributo didžiausia reikšme.
        iii. *Kiekio operatorius* grąžina reliacinę lentelę, turinčią 1-arį su n-arių skaičiumi.
        iv. *Sumos operatorius* grąžina reliacinę lentelę, turinčią 1-arį su nurodyto atributo reikšmių suma.
        v. *Vidurkio operatorius* grąžina reliacinę lentelę, turinčią tik 1-arį su nurodyto atributo mažiausia reikšme.
2. Jungčių operatoriai:
    a. Vidinių jungčių operatoriai:
        i. *Paprastoji jungtis* grąžina reliacinę lentelę, turinčią dviejų kitų lentelių visus atributus ir n-arių aibę, kurią sudaro visos galimos tų dviejų lentelių susietų n-arių kombinacijos.
        ii. *Θ jungtis* grąžina reliacinę lentelę, turinčią dviejų kitų lentelių visus atributus ir n-arių aibę, kurią sudaro visos galimos tų dviejų lentelių n-arių kombinacijos, tarp kurių atributų reikšmių galioja tam tikras sąryšis: $\{<, \leq, =, >, \geq\}$.
        iii. *Dalinė jungtis* grąžina reliacinę lentelę, turinčią pirmos lentelės visus atributus ir n-arių aibę, kurią sudaro tik tie n-ariai, kuriems egzistuoja po atskyrą n-arį kitoje lentelėje.

iv. *Antijungtis* grąžina reliacinę lentelę, turinčią pirmos lentelės visus atributus ir n-arių aibę, kurią sudaro tik tie n-ariai, kuriems neegzistuoja po atskyrą n-arį kitoje lentelėje.
v. Dalybos jungtis grąžina reliacinę lentelę, turinčią tik tuos pirmos lentelės atributus, kurių neturi antroji lentelė, ir n-arių aibę, kurią sudaro tik tie n-ariai, kurių kiekvienam egzistuoja p n-arį antroje lentelėje.

b. Išorinių jungčių operatoriai:
i. *Išorinė kairinė jungtis* - grąžina reliacinę lentelę, turinčią dviejų kitų lentelių visus atributus ir n-arių aibę, sudarytą iš visų pirmos lentelės n-arių ir susijusių antros lentelės n-arių kombinacijų; tuo atveju, kai pirmosios lentelės n-aris, neturi susieto n-ario antroje lentelėje, vietoje trūkstamų atributų reikšmių priskiriama tam tikra informacijos nebuvimo žymė.
ii. *Išorinė dešininė jungtis* - grąžina reliacinę lentelę, turinčią dviejų kitų lentelių visus atributus ir n-arių aibę, sudarytą iš visų antrosios lentelės n-arių ir susijusių pirmosios lentelės n-arių kombinacijų; tuo atveju, kai antrosios lentelės n-aris, neturi susieto n-ario pirmoje lentelėje, vietoje trūkstamų atributų reikšmių priskiriama tam tikra informacijos nebuvimo žymė.
iii. *Išorinė pilnoji jungtis* - grąžina reliacinę lentelę, turinčią dviejų kitų lentelių visus atributus ir n-arių aibę, kurią sudaro visos galimos tų dviejų lentelių n-arių (įskaitant nesusietuosius) kombinacijos, tuo atveju, kai n-aris, neturi susieto n-ario kitoje lentelėje, vietoje trūkstamų atributų reikšmių priskiriama tam tikra informacijos nebuvimo žymė.

[1 psl. 89-110; 5 psl. 61-145]

Reliacinis duomenų modelis reikalauja, kad reliacinę lentelę sudarytų n-arių gretinių aibė, t.y., kad visi n-ariai turėtų vienodą skaičių narių; šis reikalavimas užtikrina, kad kiekvienas n-aris turėtų lentelės visų atributų reikšmes. Iš aukščiau pateikto išorinių jungčių aprašymo galima pastebėti, kad galimi atvejai, kuomet reliacinėje lentelėje truks kai kurių atributų reikšmių. Šiam trūkumui žymėti, o ir kartu anksčiau paminėto modelio keliamo reikalavimo pažeidimui panaikinti, naudojama speciali žymė ω, reiškianti informacijos nebuvimą.

[1 psl. 110-111; 5 psl. 171-175]

Labai dažnai taikant reliacinį duomenų modelį praktikoje kai kurie jo terminai yra pakeičiami kitais dėl paprastumo. Reliacinė lentelė (sąryšis) yra vadinamas tiesiog *lentele*; atributai – *stulpeliais* arba *laukais*, o atributų reikšmių n-ariai – *eilutėmis*. Specialioji žymė ω dažniausiai vadinama tiesiog NULL.

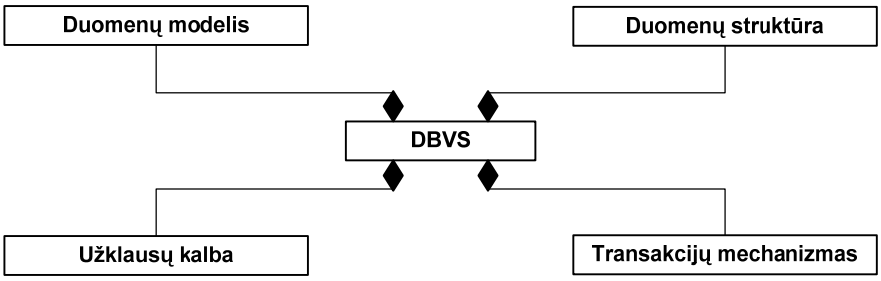
2. Reliacinių duomenų bazių valdymo sistemos

Duomenų bazių valdymo sistema (DBVS) – programinė įranga, skirta duomenų saugojimui, tvarkymui ir paieškai duomenų bazėje.

[2 psl. 2; 6 psl. 3]

Reliacinių duomenų bazių valdymo sistema (RDBVS) – DBVS, pagrįsta reliaciniu duomenų modeliu.

Pagrindiniai DBVS elementai pateikti žemiau (žr. pav. 1).



pav. 1 DBVS elementai

[1 psl. 17-18; 6 psl. 18-20]

Duomenų modelis nurodo loginę duomenų struktūrą, kuri yra taikoma duomenims, saugomiems duomenų bazėje. Plačiausiai žinomi duomenų modeliai yra pateikti psl.5. Populiariausias šiuo metu naudojamas duomenų modelis yra reliacinis.

[1 psl. 5; 6 psl. 9,12]

Duomenų struktūra (priešingai, nei duomenų modelis) nurodo kaip duomenys yra saugomi fiziškai (operatyviojoje atmintyje, failų sistemoje ir t.t.), t.y., duomenų struktūra yra žemesnio lygio abstrakcija, paprastai nematoma eiliniams duomenų bazių vartotojams.

[1 psl. 17; 6 psl. 13]

Transakcijų mechanizmo paskirtis yra užtikrinti duomenų bazėje saugomų duomenų vientisumą esant konkurenciniam jų naudojimui ar kai kurių sistemos dalių sutrikimams. Transakcijoms paprastai keliami *ACID reikalavimai*:

1. *Atomiškumas* (angl. atomicity) – įvykdomi arba visi transakcijos veiksmai arba nei vienas.
2. *Neprieštaringumas* (angl. consistency) – galimybė atšaukti pakeitimus, kurie buvo padaryti taisyklėms prieštaraujančios transakcijos metu.
3. *Atskyrimas* (angl. isoliation) – viena transakcija negali naudotis (skaityti, rašyti ir t.t.) duomenimis, kuriais tuo pačiu metu jau naudojasi kita transakcija.
4. *Ilgaamžiškumas* (angl. durability) – sėkmingai atliktos transakcijos veiksmų rezultatai turi būti nepanaikinami, t.y., turi būti galimybė atkurti visų transakcijų rezultatus (duomenų bazės būseną) po įvykusios trikties.

[1 psl. 16-17; 6 psl. 15-18,524]

Užklausų kalba leidžia vartotojams kreiptis į duomenų bazių valdymo sistemą ir prašyti atlikti tam tikrus veiksmus su pačia duomenų baze (jos struktūra) ar joje saugomais duomenimis. Kadangi duomenų bazių valdymo sistemos vartotojai per užklausų kalbą gali pasiekti duomenų bazę, užklausų kalba (DBVS dalis) yra glaudžiai susijusi ir su saugumu: vartotojų autentifikacija ir prieigos teisėmis.

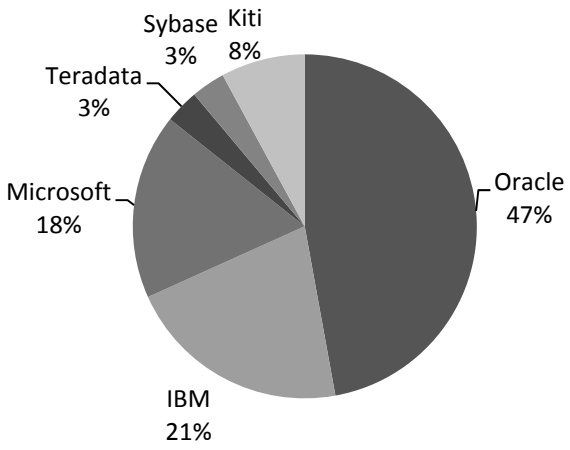
[1 psl. 12-13,18; 6 psl. 15]

SQL – struktūrizuotų užklausų kalba, skirta reliacinėje duomenų bazėje esančių duomenų paieškai ir tvarkymui, pačios reliacinės duomenų bazės struktūros kūrimui ir modifikavimui, bei prieigos teisių

valdymui. Pirmoji oficiali SQL kalbos versija (SQL-86), standartizuota 1986, turėjo priemones tik duomenų paieškai ir manipuliacijai, dėl to RDBVS gamintojai savo kalbos realizacijas papildė įvairiomis priemonėmis, leidžiančiomis SQL naudoti ir sistemos valdymo reikmėms. Vėlesnėse kalbos versijose (SQL-89, SQL-92, SQL-1999, SQL-2003, SQL-2006 ir SQL-2008) dauguma šių papildymų buvo standartizuota; nepaisant to, beveik visose RDBVS pilnai realizuotos tik pirmosios kalbos versijos ir dalis (dažniausiai su nukrypimais) vėlesniųjų. Toks negriežtas standartų laikymasis sukūrė keletą SQL dialektų: skirtingos RDBVS naudoja truputi kitokią SQL kalbą.

[1 psl. 135-137; 7 psl. 78-79]

RDBVS keliami dideli saugumo, patikimumo ir našumo reikalavimai, darantys tokių sistemų kūrimą sudėtingu ir brangiu, būtent dėl to plačiai naudojamos tik keleto gamintojų (sugebančių konkuruoti) RDBVS, nepaisant to, kad RDBVS paklausa yra didelė. Gartner Inc. firmos, specializuojančios tyrimuose IT srityje, atliktos analizės rezultatai [8] pateikti žemiau (žr. pav. 2).



pav. 2 RDBVS rinka 2006 metais

Gartner Inc. analizę atliko remdamasi finansiniu aspektu, dėl to kompanijoms ar organizacijoms, kuriančioms populiarias nemokamas (atviro kodo) RDBVS, kaip MySQL ar PostgreSQL, buvo priskirta tik maža rinkos dalis; vis dėl to iš atlikto tyrimo rezultatų matyti, kad dominuoja 3 kompanijų: Oracle, IBM ir Microsoft produktai. Žemiau pateiktos kai kurios RDBVS (žr. lent. 1).

Gamintojas	RDBVS	Licencija
Oracle Corporation	Oracle Database	Uždaro kodo
IBM	DB2	Uždaro kodo
Microsoft	SQL Server	Uždaro kodo
Sybase	Adaptive Server Enterprise	Uždaro kodo
Teradata	Teradata Database	Uždaro kodo
MySQL AB	MySQL	Atviro kodo
PostgreSQL Development Group	PostgreSQL	Atviro kodo
Ingres Corporation	Ingres Database	Atviro kodo

lent. 1 RDBVS

Iš aukščiau pateiktų duomenų matyti, kad didžiąją RDBVS rinkos dalį užimą uždarojo kodo produktai. Šis faktas yra svarbus, kadangi norint pritaikyti naujas technologijas uždaro kodo RDBVS be jų gamintojų įsikišimo, bus galima pasinaudoti tik pačių RDBVS siūlomomis priemonėmis nemodifikuojant pačių RDBVS kodo.

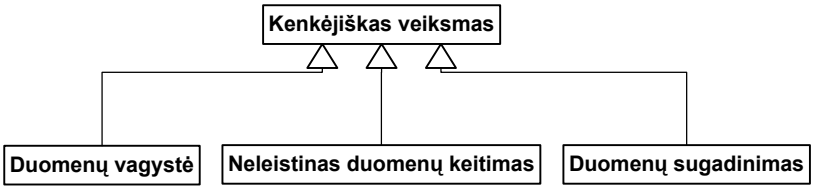
3. Reliacinių duomenų bazių apsauga

a. Saugumo daugiasluoksniškumas

Duomenų bazių saugumas – tai visuma įvairių priemonių, padedančių apsaugoti duomenų bazes nuo tyčinių kenkėjiškų veiksmų.

[1 psl. 239; 9]

Kenkėjas – asmuo, bandantis vienokiu ar kitokiu būdu tyčia atlikti veiksmą, kuriam jis neturi leidimo (teisės). Kenkėjiškų veiksmų klasifikacija pateikta žemiau (žr. pav. 3).



pav. 3 Kenkėjiškų veiksmų klasifikacija

Duomenų vagystė apsiriboja konfidencialios informacijos (pvz., banko kortelių duomenų) skaitymu; *neleistinas duomenų keitimas* pasireiškia kaip tam tikrų duomenų bazės įrašų keitimas kenkėjui palankiais (pvz., banko sąskaitos balanso pakeitimas), o *duomenų sugadinimas* – kažkokios informacijos (pvz., įrašų apie asmens padarytus nusikaltimus) sunaikinimu arba visos duomenų bazės sugadinimu, siekiant sutrikdyti kažkokią veiklą.

[1 psl. 239]

Negalima vienareikšmiškai teigti, kad kažkokia programinė įranga visiškai neturi klaidų, nes to įrodyti neįmanoma, todėl reiktų sakyti, kad toji programinė įrangą neturi nei vienos žinomos klaidos. Ta pati taisyklė galioja ir duomenų bazių sistemoms: jokia saugumo priemonė ar jų rinkinys negali vienareikšmiškai užtikrinti duomenų bazės saugumo (visada gali būti likusi kokia nors nepastebėta spraga), todėl į visas saugumo priemones būtina žiūrėti tik kaip į kliūtis kenkėjui. Pakankamas šių kliūčių kiekis ir sudėtingumas gali pareikalauti pernelyg didelio resursų kiekio iš kenkėjo, kad ataka būtų praktiškai įgyvendinama. Formalizuotai šį principą galima užrašyti taip:

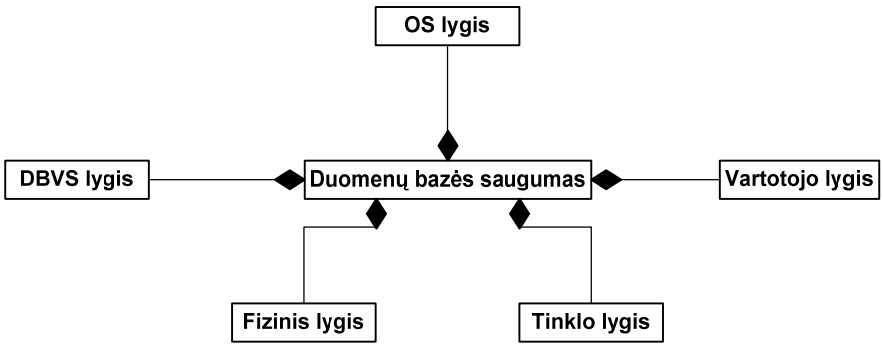
$\lim_{k\to\infty} P(k) \to 0$,

k - kliūčių skaičius;

$P(k)$ - tikimybė, kad kenkėjui pavyks įveikti sistemą.

[1 psl. 239; 10 psl. 36-38; 11; 12]

Aukščiau minėtas principas praktikoje gali būti taikomas išskaidant sistemą į dalis (sluoksnius), kurių kiekvienai būtų skirtos atskiros saugumo priemonės. Tipiškas duomenų bazės skaidymas į lygius pateiktas žemiau (žr. pav. 4).



pav. 4 Duomenų bazės saugumo lygiai

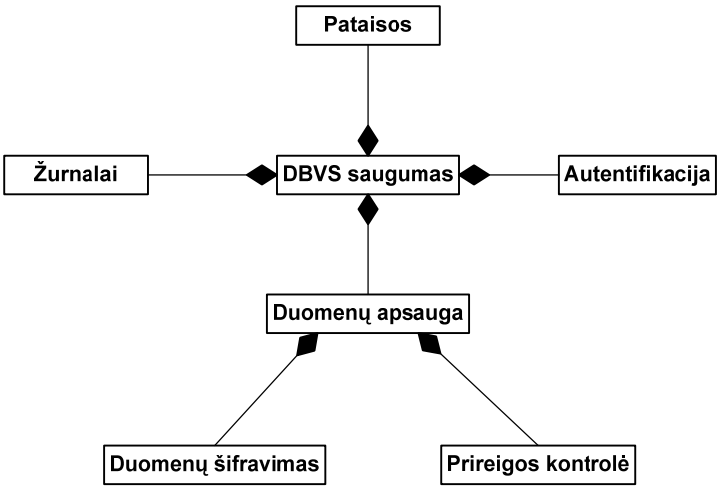
Fiziniame lygyje rūpinamasi fizine duomenų bazės apsauga: priemonėmis, neleidžiančiomis kenkėjui fiziškai pasiekti ar pakenkti su duomenų baze susijusiai įrangai. Šio lygio saugumui užtikrinti paprastai pasitelkiami įvairūs užraktai, stebėjimo kameros ir t.t.

Vartotojo lygio užduotis – apsaugoti sistemą nuo jos pačios vartotojų išnaudojimo kenkėjiškiems tikslams. Fiziniai vartotojai gali būti išnaudoti iš jų išviliojant informaciją arba net gi juos pačius sukurstant užsiimti neteisėta veikla. Pagrindinės apsaugos priemonės šiame lygyje: griežtos darbo kontrolės įvedimas, minimalių teisių sistemoje suteikimas, aiškus atsakomybės nurodymas.

Tinklo lygį sudaro priemonės, skirtos saugiam informacijos perdavimui kompiuteriniu tinklu užtikrinti. Šis lygis yra gan svarbus, kadangi dauguma šiuolaikinių duomenų bazių sistemų veikia pagal serverio ir kliento modelį. Dažniausiai naudojamos priemonės: įvairios tinklo ugniasienės ir perduodamų duomenų šifravimas.

OS lygis nagrinėja naudojamos operacinės sistemos (OS) saugumą. OS galima laikyti kaip tarpinę programinę įrangą tarp aparatūrinės įrangos ir aukštesnio lygio programinės įrangos (DVBS). OS atsakinga tiek už aparatūros, tiek ir už programinės įrangos valdymą, dėl to yra svarbi saugumo grandis. Pagrindinės jos apsaugos priemonės: vykdomų procesų izoliacija, minimalių procesų teisių suteikimas, failų sistemos prieigos teisės ir šifravimas.

DBVS lygis – tai pačios DBVS saugumo priemonės, skirtos tiesioginiai duomenų bazės apsaugai. DBVS lygis yra smulkiau detalizuotas žemiau (žr. pav. 5).



pav. 5 DBVS saugumo lygis

Pataisos – priemonė plačiai taikoma ne tik DBVS, bet ir kitoje programinėje įrangoje, užtikrinanti greitą atrastų programinių klaidų ir saugumo spragų šalinimą;

Autentifikacija – DBVS saugumo mechanizmas patikrinantis vartotojo tapatybę (slaptažodžiu, sertifikatu ir k.t.), bei susiejantis vartotoją su jam suteiktomis privilegijomis sistemoje;

Žurnalai – vartotojų atliekamų veiksmų registravimas ir analizavimas, įskaitant automatinį sistemos reagavimą į aptiktus pažeidimus;

Duomenų apsauga – priemonės tiesiogiai saugančios duomenis nuo neteisėto jų panaudojimo ar manipuliavimo. Ši apsauga gali būti realizuojama taikant prieigos kontrolę (leidžiant prie duomenų prieiti per DBVS sistemą tik autentifikuotiems vartotojams, turintiems tinkamas privilegijas) ir naudojant duomenų šifravimą, kad duomenimis nebūtų galima pasinaudoti apėjus DBVS (pvz., analizuojant failų sistemą, kurioje saugoma DB).

b. Saugumo tipai

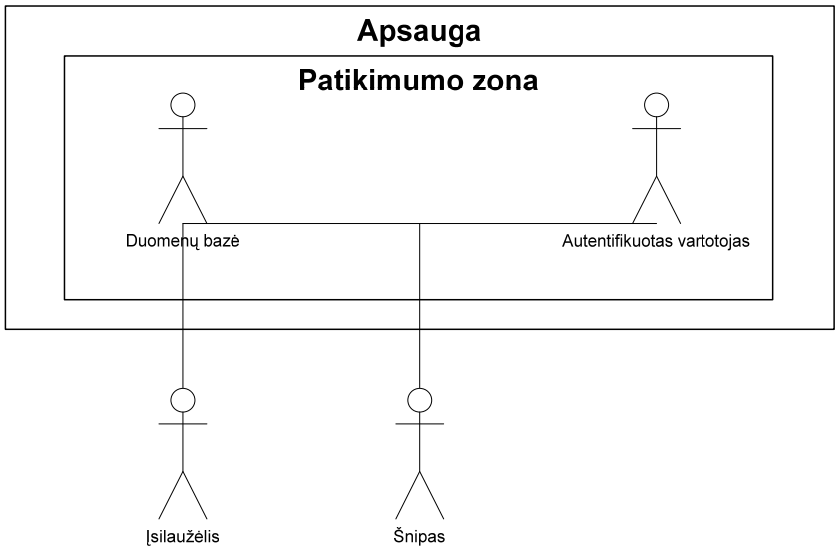
i. Vienašalio saugumo modelis

Vienašalio saugumo modelis – modelis, kuriame tam tikrus sistemos elementus sieja vienpusis nepasitikėjimas.

[9]

Vienašališkumą reiktų suprasti kaip saugumo priemonių taikymą tik vienai iš sąveikaujančių pusių. Paprasčiausias vienašalio saugumo pavyzdys yra banko kasa, apsaugota grotomis, kurios skirtos užtikrinti kasininko, bet ne kliento saugumą.

Informacinėse sistemose vienašališko saugumo modelis yra labai plačiai taikomas dėl savo natūralumo (užrakinama tam, kad apsaugoti nuo išorės arba tam, kad apsaugoti išorę, bet ne tam, kad būtų apsaugota ir tas, kas užrakinta, ir išorė kartu) ir paprastos realizacijos. Didžioji dauguma programinės įrangos, įskaitant ir DBVS remiasi šiuo modeliu. Vienašalio saugumo modelio taikymo DBVS pavyzdys pateiktas žemiau (žr. pav. 6).



pav. 6 Vienašalis DBVS saugumo modelis

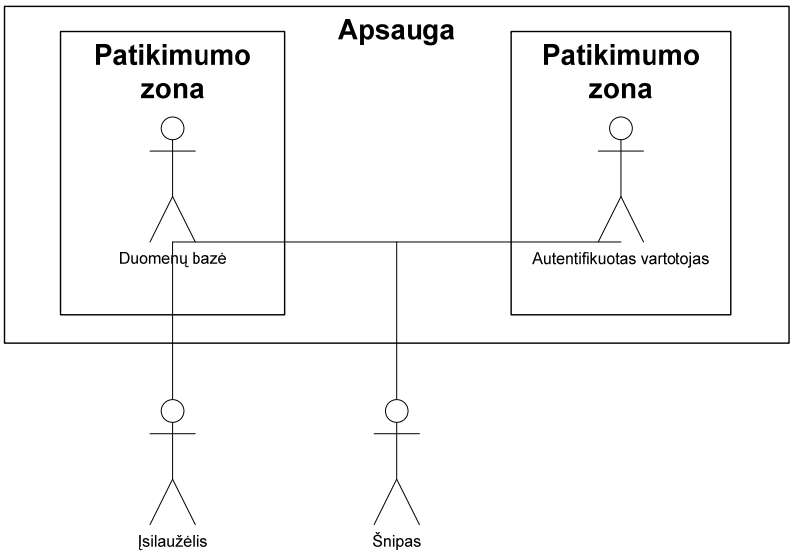
DBVS atveju sistemos elementais yra pati DBVS ir autentifikuoti (būtent autentifikuoti) jos vartotojai. Įsilaužėlis, bandantis neteisėtai patekti į DBVS, ir šnipas, bandantis neteisėtai stebėti perduodamus duomenis, nepriklauso šiai sistemai, t.y., jie – kenkėjai, trukdymas kuriems patekti į sistemą ir yra saugumo užduotis. Vienašališkumas pasireiškia tuo, kad autentifikuotas vartotojas visiškai pasitiki DBVS, tačiau DBVS tuo pačiu neatsako: pritaikoma prieigos kontrolė, sekami ir analizuojami veiksmai ir t.t.

ii. Daugiašalio saugumo modelis

Daugiašalio saugumo modelis – modelis, kuriame tam tikrus sistemos elementus sieja abipusis (daugiašalis) nepasitikėjimas.

[9]

Daugiašalį saugumą galima iliustruoti per milijonierių uždavinį. Tarkime yra keletas milijonierių, kurie nenori atskleisti savo turimo turto dydžio, tačiau nori išsiaiškinti, kuris iš milijonierių yra pats turtingiausias. Iškyla problema: tarpusavyje norima bendradarbiauti (išsiaiškinti, kas turtingiausias), tačiau nėra tarpusavio pasitikėjimo (negalima apsikeisti informacija). Vienas iš tokio uždavinio sprendimo būdų gali būti patikimo tarpininko įvedimas: milijonieriai jam pasako savo turimo turto dydį, ir jis paskelbią turtingiausio milijonieriaus vardą neatskleisdamas, kiek turto kuris milijonierius turi. Daugiašalio saugumo modelio taikymo DBVS pavyzdys pateiktas žemiau (žr. pav. 7).



pav. 7 Daugiašalio DBVS saugumo modelis

Įsilaužėlio ir šnipo padėtis yra analogiška jų padėčiai vienašalio saugumo modelyje. Autentifikuoto vartotojo situacija žiūrint iš DBVS taško irgi tokia pati: DBVS taiko prieigos kontrolę ir seka vartotojo veiksmus. Pagrindinis skirtumas tarp vienašalio ir daugiašalio DBVS saugumo modelių yra autentifikuoto vartotojo pasitikėjimas pačia DBVS; daugiašaliame DBVS saugumo modelyje vartotojas nepasitiki pačia DBVS. Vartotojo nepasitikėjimo DBVS priežasčių gali būti daug; kai kurios iš jų pateiktos skyrelyje „Daugiašalio duomenų bazės saugumo modelio aktualumas“ (žr. 14 psl.).

II. Dalykinės srities analizė

4. Daugiašalio duomenų bazės saugumo modelio aktualumas

Nors daugiašalis duomenų bazės modelis gali atrodyti ir nenatūralus, tačiau praktikoje pasitaiko situacijų, kuomet jis yra būtinas:

1. *Susikompromitavęs administratorius.* Administratorius sistemoje pasižymi tuo, kad jis iš visų vartotojų turi aukščiausias teises. Žinoma, būtent dėl to, jis turi priėjimą prie visų duomenų bazėje esančių duomenų. Nors dažniausiai yra atskiriamos sistemos administratoriaus ir saugumo administratoriaus rolės, tačiau tai tik minimaliai sumažina pavojų, kuomet vienas iš administratorių tampa kenkėju (pvz., nori pavogti duomenų bazėje saugomą informaciją).
2. *Interesų konfliktas.* Tarkime egzistuoja kompanija, užpatentavusi nemažai išradimų ir sauganti patentų informaciją duomenų bazėje. Ši duomenų bazė yra viešai prieinama norintiems susipažintu su patentais. Kitai kompanijai darant užklausas duomenų bazėje apie ją dominančius patentus, pirmoji tai matydama gali padaryti išvadas apie antrosios veiklos planą ir pagal tai pakoreguoti veiklą antrosios kompanijos nenaudai.
3. *Serverio nuoma*. Ne visi gali sau leisti turėti nuosavą dedikuotą serverį ir samdyti priežiūros personalą, todėl serverių nuoma yra populiarus šiandienos reiškinys. Iš to, kad informacijos saugojimo vieta ir priežiūra priklauso ne informacijos savininkui, iškyla nepasitikėjimo ir saugumo problema.
4. *Duomenų gavyba*. Kartais svarbu apsaugoti individualių įrašų informaciją, leidžiant pasinaudoti tik sudėtine informacija; pvz. laikyti kiek koks asmuo uždirba paslaptyje, tačiau leisti sužinoti visų asmenų uždarbio vidurkį.

Iš aukščiau pateiktų pavyzdžių galima padaryti išvada, kad duomenų bazių valdymo sistema galima ne visada pasitikėti, todėl daugiašalio duomenų bazės saugumo principą įmanoma performuluoti kaip reikalavimus duomenų bazės sistemai:

1. DBVS turi saugoti informaciją.
2. DBVS turi atlikti veiksmus (skaičiavimus) su informacija.
3. DBVS negali „suvokti“ joje saugomos informacijos.

5. Moksliniai darbai, susiję su daugiašaliu duomenų bazių saugumu

Darbe [13] pirmą kartą panaudotas terminas „Hipokratinė duomenų bazė“. Nors autoriai ir rėmėsi pacientų duomenų bazių pavyzdžiais, tačiau hipokratinės duomenų bazės sąvoką reiktų suprasti, kaip duomenų bazę, kurioje akcentuojamas informacijos konfidencialumas. Šiame minėtame darbe pateiktos pagrindinės taisyklės ir galimi metodai tokių duomenų bazių realizacijai.

Vėlesnieji darbai gali būti suskirstyti į 2 grupes: konfidencialumo pasiekimas per detalias prieigos teises ir užklausų vykdymas neiššifruojant duomenų.

Šalutinių duomenų, turinčių panašų pasiskirstymą, kaip tikrieji, generavimas duomenų gavybai aptariamas [14]. Duomenų prieigos teisių skaidymo formalizavimas ir duomenų slėpimas nagrinėjamas [15]. Duomenų žymėjimo pagal jų paskirtį ir prieigos teisių paskyrimo remiantis tomis žymėmis metodai pateikti [16].

SQL užklausų vykdymas neiššifravus duomenų ir jų formalus skaidymas siekiant, minimizuoti skaičiavimus, nagrinėjami [17]. Algoritmas, leidžiantis išlaikyti eiliškumą užšifravus duomenis, pateiktas [18]. Agregatinių funkcijų skaičiavimo neiššifruojant duomenų metodas duotas [19].

Pirmosios darbų grupės siūlomi metodai nėra tinkami daugiašaliam duomenų bazių saugumui užtikrinti, kadangi smulkių prieigos teisių taikymu rūpinasi pati DBVS, o šios prieigos teisės tik apribojа prieigą prie duomenų DBVS vartotojams, o ne pačiai DBVS. Antrosios grupės darbuose siūlomos priemonės leidžia DBVS atlikti duomenų paiešką ir skaičiavimus neiššifruojant duomenų, t.y. jų „nesuvokiant“, todėl šios priemonės tinkamos daugiašaliui reliacinės duomenų bazės saugumui užtikrinti.

6. Metodai, leidžiantys atlikti užklausas neiššifravus duomenų

a. Pailerio kriptosistema

Pailerio kriptosistema – asimetrinis kriptografinis algoritmas, sukurtas Paskalio Pailerio.[20]

Raktų generavimas:

1. Parenkami du dideli pirminiai skaičiai p ir q.
2. Apskaičiuojama $n = pq$ ir $\lambda = mbk(p-1, q-1)$, kur $mbk(x, y)$ yra skaičių x ir y mažiausias bendras kartotinis.
3. Parenkamas atsitiktinis skaičius g toks, kad $dbd\left(L\left(g^{\lambda} \bmod n^2, n\right)\right) = 1$, kur $dbd(x, y)$ yra skaičių x ir y didžiausias bendrasis daliklis, o $L(u) = \frac{u-1}{n}$.
4. Apskaičiuojama $\mu = \left(L\left(g^{\lambda} \bmod n^2\right)\right)^{-1} \bmod n$.

Viešasis raktas: (n, g).

Privatusis raktas: (λ, μ).

Šifravimas:

1. $m < n$, kur m - atviras tekstas.
2. Parenkamas atsitiktinis skaičius r toks, kad $r < n$.
3. Apskaičiuojamas slaptaraštis $c = g^m \cdot r^n \bmod n^2$.

Iššifravimas:

1. c – slaptaraštis.
2. Apskaičiuojamas atviras tekstas $m = L\left(c^{\lambda} \bmod n^2\right) \cdot \bmod n$.

Pailerio kriptosistema pasižymi šiomis homomorfinėmis savybėmis:

1. Atviro teksto sudėtimi:
    a. $D(E(m_1, r_1) \cdot E(m_2, r_2)\ mod\ n^2) = m_1 + m_2\ mod\ n.$
    b. $D(E(m_1, r_1) \cdot g^{m_2}\ mod\ n^2) = m_1 + m_2\ mod\ n.$
2. Atviro teksto sandauga: $D(E(m_1, r_1)^k\ mod\ n^2) = km_1\ mod\ n.$

čia: D - iššifravimo funkcija, E - šifravimo funkcija, k – konstanta.

Atviro teksto sudėties savybė yra ypač svarbi, kadangi leidžia atviro teksto sudėtį pakeisti slaptaraščio daugyba. Pritaikius šią kriptosistemą duomenų bazių apsaugai, būtų galima pasiekti, kad DBVS atlikinėtų sudėties veiksmus neiššifruodama duomenų, be to, slaptaraščio daugybos metu gautas atsakymas taip pat butų saugus, kadangi jis sumos prasmę įgautų tik, kai būtų iššifruotas.

b. OPES

OPES (angl. Order Preserving Encryption Scheme) – šifravimo algoritmas, pasiūlytas [18]. Pagrindinė algoritmo idėja yra remiantis nešifruotų reikšmių ir norimu šifruotų reikšmių skirstiniais pakeisti nešifruotais reikšmes taip, kad išliktų eiliškumas.

Raktai

OPES raktas – pora $\{K^f, K^c\}$.

Struktūrą K sudaro keletas B tipo elementų, reiškiančių intervalus, į kuriuos yra suskaidyta tam tikra duomenų sritis.

Elementai B yra sudaryti iš šių dalių:

1. a - intervalo pradžia.
2. b - intervalo pabaiga.
3. s - kvadratinis koeficientas.
4. z - mastelio faktorius.

K struktūros pildymas:

1. Iš pasirinktos reikšmių srities pagal pasirinktą skirstinį sugeneruojama reikšmių imtis
    a. Reikšmės K^f generuojamos pagal nešifruotų duomenų skirstinį.
    b. Reikšmės K^c generuojamos pagal norimą šifruotų duomenų skirstinį.
2. Reikšmių imtys rekursyviai skaidomos į intervalus:
    a. Apskaičiuojamas intervalo skirstinys – tiesė, jungianti taškus (x_0, p_0) ir (x_{n+1}, p_{n+1}), kur x_0 - pirmoji imties reikšmė, pakliūnanti į intervalą; x_{n+1} – pirmoji gretimo intervalo imties reikšmė; p_0 – pirmosios reikšmės tikimybė pagal imties skirstinį; p_{n+1} – gretimo intervalo pirmosios reikšmės tikimybė pagal imties skirstinį.
    b. Jei intervale yra daugiau nei iš anksto nustatytas skaičius (pvz., 10) imties reikšmių, tuomet apskaičiuojama kiekvienos reikšmės tikimybė remiantis intervalo skirstiniu (anksčiau sudaryta tiese). Intervalas perskiriamas pusiau į du kitus intervalus ties

reikšme, kurios tikimybė pagal imties skirstinį yra labiausiai nukrypusi nuo tikimybės pagal intervalo skirstinį.

c. Kiekvienam intervalui apskaičiuojamas kvadratinis koeficientas: $s = \frac{q}{2r}$, kur q ir r randami iš intervalo skirstinio lygties $qp + r$, kur p - intervalui priklausanti reikšmė; struktūroms K^f ir K^c kvadratiniai koeficientai žymimi atitinkamai s^f ir s^c.

3. Apskaičiuojami mastelio faktoriai z:

a. Struktūrai K^f: $z_i^f = \frac{R^f n_i^f}{s_i^f \left(w_i^f\right)^2 + w_i^f}$, kur

i. w_i^f - intervalo plotis;

ii. $R^f = max\left[\widehat{w}_i^f\right],\ i = 1, \dots m$, m – intervalų skaičius;

iii. $\widehat{w}_i^f = \hat{z}_i^f \left(s_i^f \left(w_i^f\right)^2 + w_i^f\right)$;

iv. $\hat{z}_i^f = \begin{cases} 2 & s_i^f \geq 0 \\ \frac{2}{1 + s_i^f \left(2w_i^f - 1\right)} & s_i^f < 0 \end{cases}$.

b. Struktūrai K^c: $z_i^c = \frac{R^c n_i^c}{s_i^c (w_i^c)^2 + w_i^c}$, kur

i. w_i^c - intervalo plotis;

ii. $R = min\left[\frac{\widehat{w}_i^c}{n_i^c}\right],\ i = 1, \dots k$, k – intervalų skaičius;

iii. $\widehat{w}_i^c = \hat{z}_i^c (s_i^c (w_i^c)^2 + w_i^c)$;

iv. $\hat{z}_i = \begin{cases} \frac{1}{2\left(1 + s_i^c (2w_i^c - 1)\right)} & s_i^c > 0 \\ \frac{1}{2} & s_i^c \leq 0 \end{cases}$.

Pagalbinės funkcijos:

a. $M_i^t(x) = z_i^t (s_i^t x^t + x)$;

b. $(M^t)^{-1}(x) = \frac{-z_i^t \pm \sqrt{(z_i^t)^2 + 4 z_i^t s_i^t x}}{2 z_i^t s_i^t}$.

PASTABA: t nurodo struktūrą.

Mastelio koeficientas:

$$L = \frac{\sum_{j=1}^{m} \mathrm{M_j^f}\left(\mathrm{w_j^f}\right)}{\sum_{j=1}^{i-1} \mathrm{M_j^c}\left(\mathrm{w_j^c}\right)}$$

Šifravimas:

1. $f = f_{min} + \sum_{j=1}^{i-1} \mathrm{M_j^f}(\mathrm{w_j^f}) + \mathrm{M_i^f}(p - p_{min} - \sum_{j=1}^{i-1} \mathrm{w_j^f})$;
2. $c = c_{min} + \sum_{j=1}^{i-1} Lw_j^c + (M_i^c)^{-1}(f - f_{min} - \sum_{j=1}^{i-1} \mathrm{LM_j^c}(\mathrm{w_j^c}))$.

Iššifravimas:

1. $f = f_{min} + \sum_{j=1}^{i-1} \mathrm{LM_j^c}(\mathrm{w_j^c}) + M_i^c(c - c_{min} - \sum_{j=1}^{i-1} \mathrm{L}w_j^c)$;
2. $p = p_{min} + \sum_{j=1}^{i-1} \mathrm{w_j^f} + (M_i^f)^{-1}(f - f_{min} - \sum_{j=1}^{i-1} \mathrm{M_j^f}(\mathrm{w_j^f}))$.

čia

- p - nešifruota reikšmė;
- c - šifruota reikšmė;
- p_{min} – mažiausia galima nešifruotų duomenų reikšmė;
- f_{min} – mažiausia galima tarpinių duomenų reikšmė;
- c_{min} – mažiausia galima šifruotų duomenų reikšmė.

III. Daugiašalis saugumo modelis, pagrįstas šifravimu

1. Trumpas modelio aprašymas

Šiame skyriuje aprašytas sudarytas reliacinės duomenų bazės saugumo modelis, leidžiantis duomenų bazėje saugoti skaitinius duomenis slepiant jų tikrąsias reikšmes nuo RDBVS.

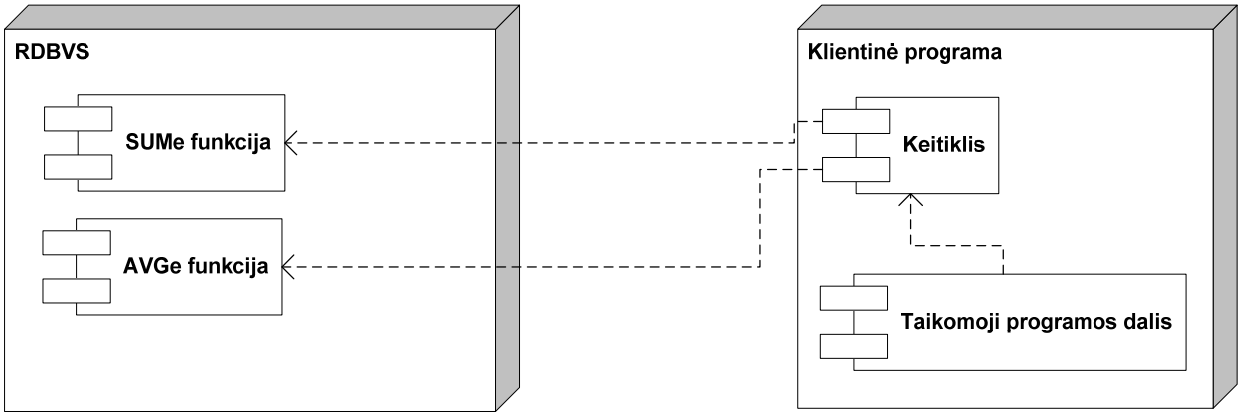
Daugiašalis saugumas pasiekiamas naudojant dvigubą duomenų šifravimą (taikant OPES ir Pailerio algoritmus), kad būtų galima kartu panaudoti šių dviejų algoritmų svarbiausias savybes: duomenų eiliškumo nepraradimą dėl šifravimo ir duomenų sumos pakeitimą šifruotų duomenų sandauga.

Naudojant specialią klasę klientinės programos užklausoms ir RDBVS grąžinamiems rezultatams modifikuoti, pasiekiama, kad skaitinių duomenų paieška, sumos, bei vidurkio agregatinės funkcijos būtų vykdomos RDB valdymo sistemoje neiššifruojant duomenų, o rezultatai į klientinę programą perduodami jau iššifruoti.

Tolimesniuose šio skyriaus poskyriuose šis sudarytas modelis yra detalizuojamas smulkiau.

2. Sistemos elementai

Modelį galima suskaidyti į keletą pagrindinių elementų (žr. pav. 8), kurių dalis priklauso klientinei programai, o dalis RDBVS. Gali pasirodyti, jog naujų elementų būvimas RDBVS reiškia RDBVS modifikavimą, tačiau iš tiesų taip nėra. RDBVS priklausantys elementai nereikalauja RDBVS programinio kodo keitimo, kadangi paprastai jie gali būti realizuoti arba kaip SQL funkcijos ar procedūros arba kaip vartotojų funkcijų moduliai.



pav. 8 Modelio elementai

Taikomoji programos dalis – tipiška klientinės programos dalis, formuojanti užklausas RDBVS ir naudojanti užklausų rezultatus. Paprastai (ne šiame modelyje) jos suformuotos užklausos būna perduodamos tiesiogiai RDBVS, tačiau šiame modelyje visos užklauso ir rezultatai yra perduodami ne tiesiogiai, o per keitiklį.

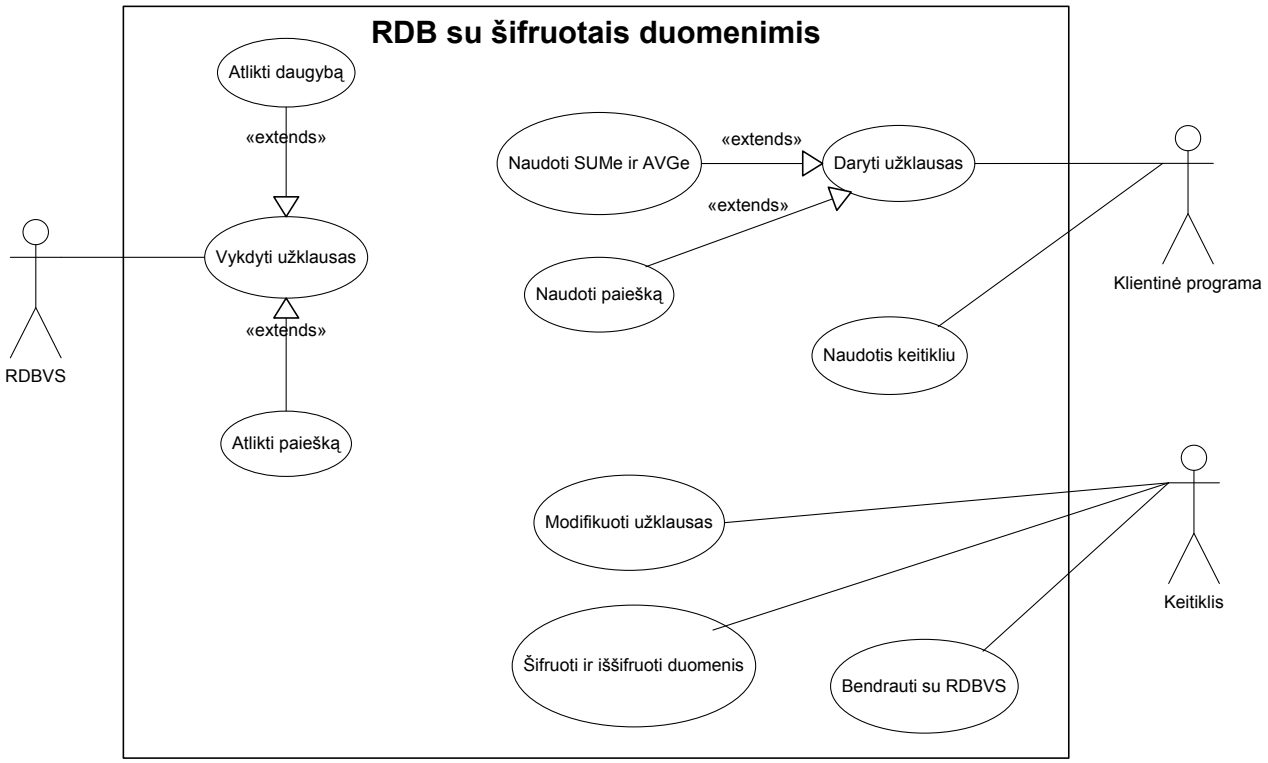
Keitiklis – klasė ar modulis, per kurį vyksta mainai tarp RDBVS ir Taikomosios programos dalies. Šis elementas atsakingas už užklausų ir rezultatų modifikavimą.

SUMe – vartotojo sukurta agregatinė RDBVS funkcija, sudauginanti tam tikro reliacinės lentelės stulpelio reikšmes moduliu n. Ši funkcija naudojama šifruotų reikšmių daugybai, iššifravus kurios rezultatą gaunama reikšmių suma.

AVGe - vartotojo sukurta agregatinė RDBVS funkcija, sudauginanti tam tikro reliacinės lentelės stulpelio reikšmes moduliu n ir pakelianti gautą sandaugą reikšmių skaičiaus laipsniu moduliu n^2. Iššifravus gautą rezultatą gaunamas reikšmių vidurkis.

3. Sistemos elementų veiklos sritys

Nors poskyryje „Sistemos elementai“ (žr. psl. 19) pateikti 4 elementai, tačiau nagrinėjant jų veiklos sritis, yra naudinga kai kuriuos iš jų apjungti. Žemiau pateiktoje diagramoje (žr. pav. 9) nurodoma tik 3 elementų: RDBVS (įskaitant SUMe ir AVGe), Klientinės programos (taikomosios dalies) ir keitiklio, veikla.



pav. 9 Sistemos elementų veiklos sritys

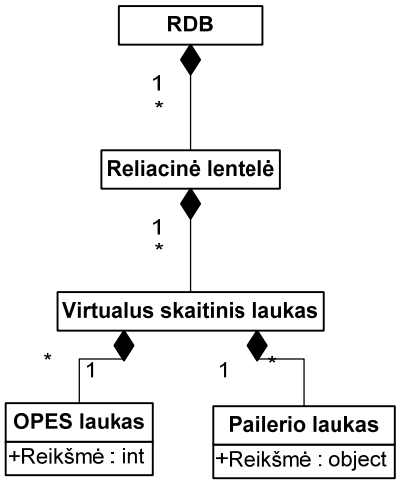
Klientinės programos (taikomosios programos dalies) veikla apsiriboja užklausų formavimu, naudojant SumE, AVGe funkcijas, paieškos operatorius (=,>,<,<=,>=), ir rezultatų naudojimu. Tiesa, ji tai atlieka naudodamasi (kreipdamasi į) keitiklį. Duomenų šifravimu ir iššifravimu klientinė programos dalis nesirūpina.

Keitiklis atlieka užklausų modifikavimą: pakeičia funkcijų vardus, užšifruoja duomenis, perduoda užklausą RDBVS ir gavęs rezultatus, juos iššifruoja.

RDBVS lieka standartinių užklausų, bei papildomų funkcijų SUMe ir AVGe. Reiktų pabrėžti, kad RDBVS negali atlikti nei šifravimo, nei iššifravimo (tą gali padaryti tik keitiklis) – tai daugiašalio RDB saugumo esmė.

4. Duomenų struktūra

Šiame modelyje naudojamas dvigubas duomenų šifravimas, leidžiantis atlikti tiek paiešką, tiek ir pritaikyti sumos ir vidurkio agregatines funkcijas neiššifravus duomenų. Dėl dvigubo šifravimo būtina duomenų bazėje išsaugoti tuos pačius duomenis 2 kartus (taikant skirtingus algoritmus). Duomenų struktūra pateikta žemiau (žr. pav. 10).



pav. 10 Duomenų struktūra

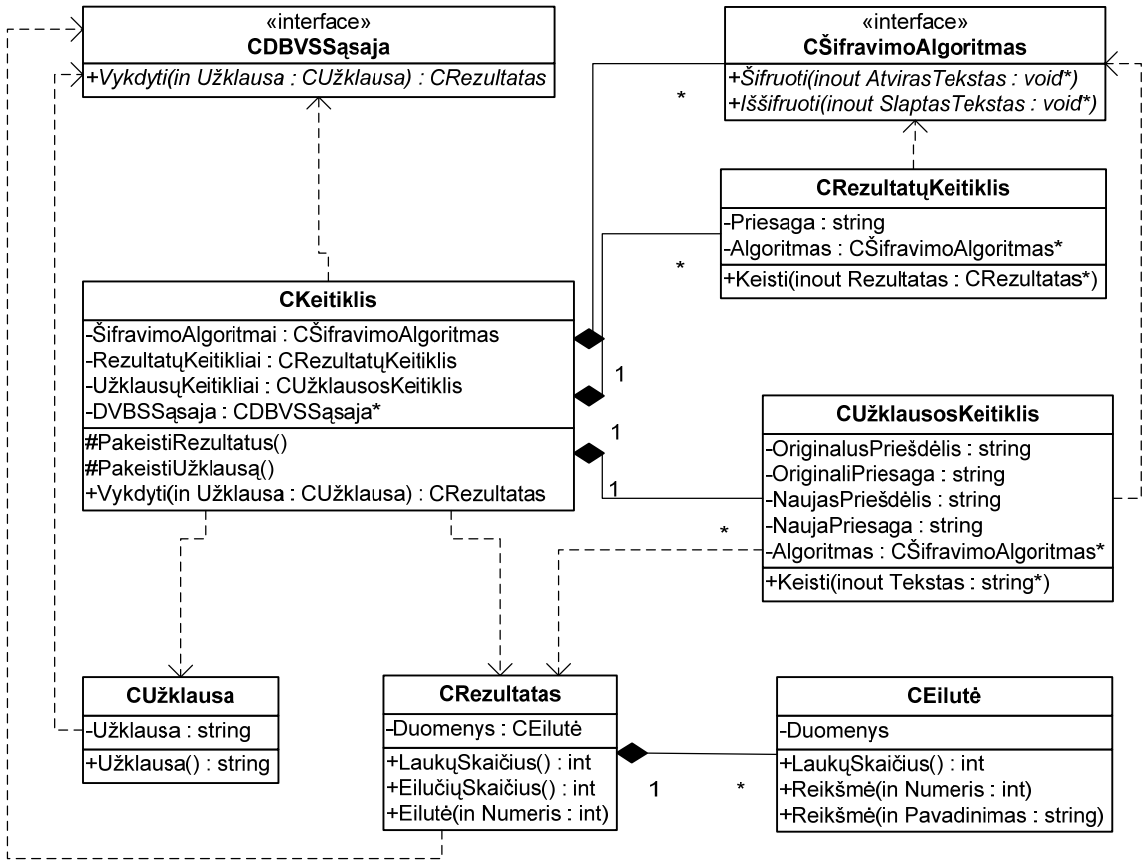
Taikant modelį nešifruotai reliacinei duomenų bazei, vietoje kiekvieno skaitinio reliacinės lentelės lauko, kurį norima apsaugoti, būtina sukurti du 2 laukus. Pirmasis laukas saugo reikšmę, užšifruotą OPES algoritmu, o antrasis – reikšmę, užšifruotą Pailerio algoritmu. Kadangi abu laukai saugo tuos pačius duomenis (tik kitoje formoje), jų abiejų porą galima laikyti virtualiu lauku.

OPES lauko dydis priklauso nuo pasirinktų skirstinių, tačiau blogiausiasis atvejis gali būti įvertintas kaip lauko padidėjimas pagal formulę $\log\frac{g^p_{max}}{g^p_{min}} + \log\frac{g^c_{max}}{g^c_{min}}$, kur g^p_{max} - didžiausias atstumas tarp rūšiuotų gretimų nešifruotų reikšmių, g^p_{min} – mažiausias atstumas tarp rūšiuotų gretimų nešifruotų reikšmių, g^c_{max} - didžiausias atstumas tarp rūšiuotų gretimų šifruotų reikšmių, g^c_{min} – mažiausias atstumas tarp rūšiuotų gretimų šifruotų reikšmių. Praktiškai blogiausiu atveju OPES lauko dydis padidės ne daugiau nei 64 bitais[18].

Pailerio lauko dydis tiesiogiai priklauso nuo naudojamo rakto dydžio. Šiuo metu yra laikoma, jog 1024 bitų arba ilgesni raktai yra saugūs, todėl Pailerio laukas turėtų būti BLOB (angl. Binary Large Object) tipo, leidžiančio saugoti dideles reikšmes.

5. Keitiklis

Keitiklis, atsakingas už užklausų ir rezultatų modifikavimą, gali būti suskaidytas į smulkesnes dalis. Jo skaidymas į klases pateiktas žemiau (žr. pav. 11).



pav. 11 Keitiklio klasės

CDBVSSąsaja – interfeisas, per kurį yra vykdomi duomenų manai tarp Keitiklio ir konkretaus tipo RDBVS. Šis interfeisas realizuojamas parašius specialią klasę, galinčią dirbti su norima RDBVS.

CŠifravimoAlgoritmas – interfeisas, aprašantis sąsają per kurią Keitiklis naudosis reikalingais šifravimo algoritmais. Šiame modelyje yra būtinos 3 šio interfeiso realizacijos: OPES, Pailerio ir dar viena pagalbinė.

CRezultatųKeitiklis – klasė, modifikuojanti iš RDBVS gautus rezultatus. Rezultatai modifikuojami pritaikant nurodytą šifravimo algoritmą kiekvienam grąžintos reliacinės lentelės laukui, turinčiam nurodytą priesagą pavadinime. Šios klasės objektų keitiklyje privalo būti bent 2, kadangi modelyje naudojami 2 šifravimo algoritmai.

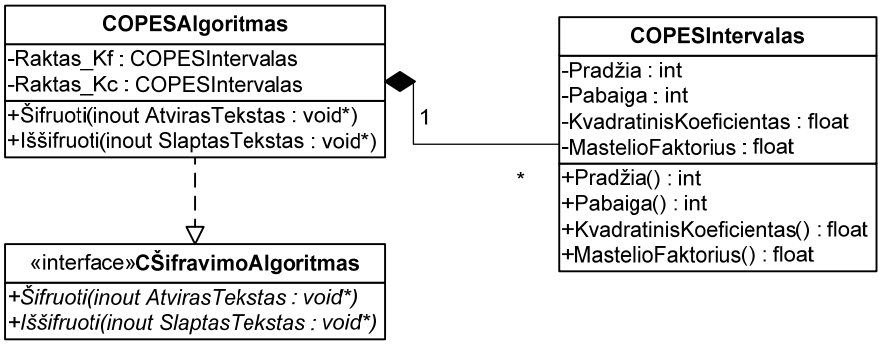
CUžklausosKeitiklis - klasė, modifikuojanti suformuotą užklausą prieš ją išsiunčiant į RDBVS. Modifikavimas vykdomas ieškant nurodytų priešdėlių ir priesagų SQL užklausoje ir jas keičiant naujomis, bei pritaikant šifravimo algoritmą (jei jis nurodytas) šakniai. Šios klasės objektų turi būti keletas norint realizuoti šį modelį.

CUžklausa – klasė, kurios objektas reiškia konkrečią užklausą.

CRezultatas – klasė, kurios objektas reiškia konkrečius rezultatus – reliacinę lentelę. Ši klasė yra sudaryta iš 0 arba daugiau CEilučių, reiškiančių reliacinės lentelės eilutes.

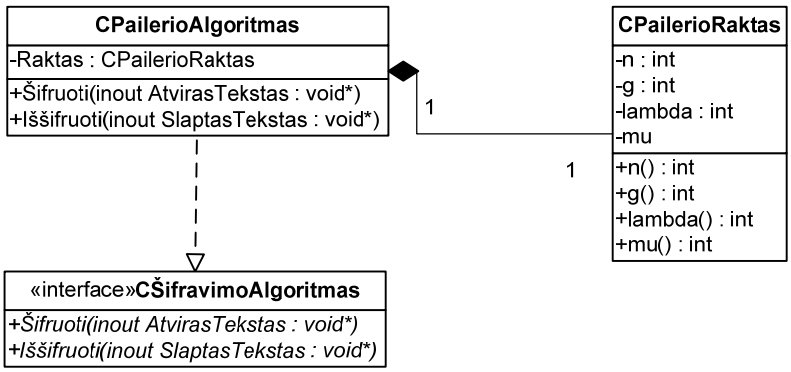
CKeitiklis yra pagrindinė Keitiklio klasė, per kurią klientinė programos dalis siunčia užklausas ir gauna rezultatus. Šioje klasėje saugoma keletas CŠifravimoAlgoritmų, CUžklausųKeitiklių ir CRezultatųKeitiklių, reikalingų tinkamam užklausų ir rezultatų modifikavimui.

Kaip jau buvo minėta aukščiau, Keitiklyje reikalingos 3 CŠifravimoAlgoritmo realizacijos. Jų diagramos pateiktos žemiau (žr. pav. 12, pav. 13 ir pav. 14).



pav. 12 CŠifravimoAlgoritmo realizacija COPESAlgoritmas

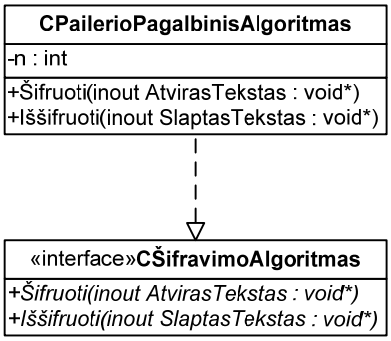
OPES algoritmo raktą sudaro keletas COPESIntervalas objektų, kuriuose saugoma kiekvieno intervalo pradžia, pabaiga, kvadratinis koeficientas ir mastelio faktorius. Šifruojant ir iššifruojant nustatomas intervalas, kuriam priklauso šifruojama ar iššifruojama reikšmė ir pritaikomas atitinkamas kvadratinis koeficientas ir mastelio faktorius.



pav. 13 CŠifravimoAlgoritmo realizacija CPailerioAlgoritmas

CPailerioAlgoritmo raktas yra žymiai paprastesnis nei COPESAlgoritmo, kadangi jį sudaro tik 4 skaičiai: n, g, lambda, mu.

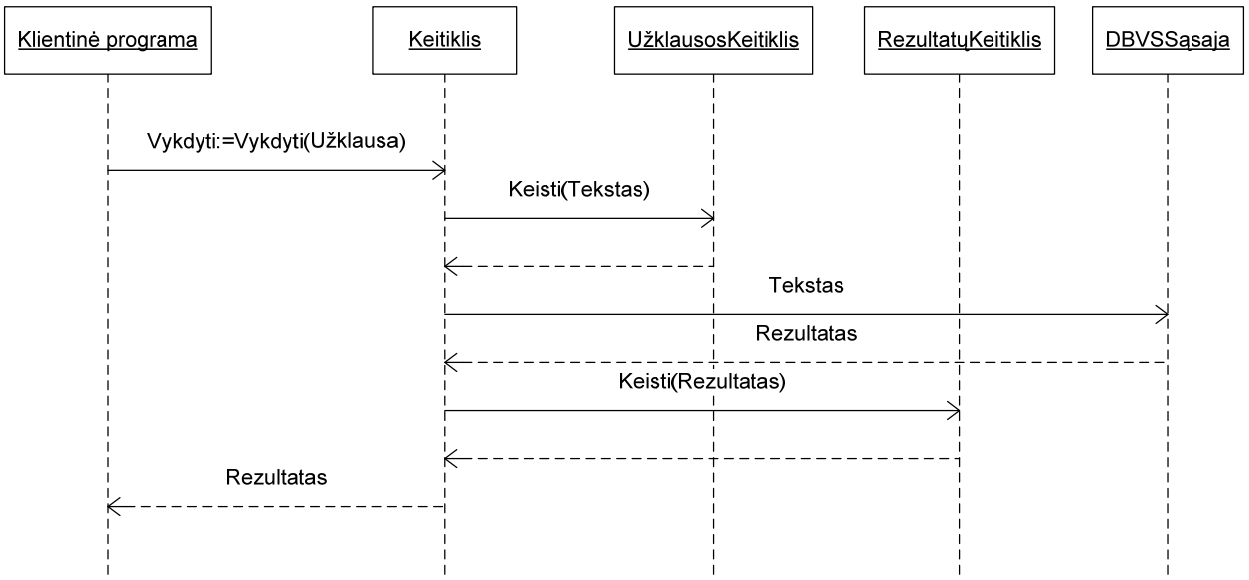
Tiek COPESAlgoritmo, tiek ir CPailerioAlgoritmo veikimo principas paprastas: metodas „Šifruoti“ užšifruoja nurodytą reikšmę, o metodas „Iššifruoti“ iššifruoja; tačiau su CPailerioPagalbinisAlgoritmas yra kiek kitaip.



pav. 14 CŠifravimoAlgoritmo realizacija CPailerioPagalbinisAlgoritmas

CPailerioPagalbinioAlgoritmo paskirtis yra ne šifruoti, o perdaryti gautą reikšmę (tekstą). Norint pritaikyti sumos ir vidurkio agregatines funkcijas RDBVS pusėje, būtina į RDBVS perduoti ir Pailerio viešojo rakto dalį n, kuri naudojama skaičiavimuose. CPailerioPagalbinisAlgoritmas modifikuoja gautą tekstą prie jo pridėdamas n reikšmę (pvz., „laukas“ pakeičiamas į „laukas, n“, kur n - reikšmė).

6. Užklausų vykdymas



pav. 15 Užklausos vykdymas

Užklausos vykdymas prasideda nuo to, kad klientinė programa suformuoja reikalingą SQL užklausą ir ją perduoda Keitikliui (konkrečiai CKeitikliui).

CKeitiklis gautą užklausos tekstą paeiliui perduoda kiekvienam užklausų keitikliui (CUžklausosKeitiklis). Užklausų keitikliai, remdamiesi savo parametrais, modifikuoja (ir šifruoja, jei reikia) gautą SQL užklausos tekstą. Reikalingi užklausų keitikliai ir jų parametrai pateikti toliau (žr. lent. 2).

Nr.	Originalus priešdėlis	Originali priesaga	Naujas priešdėlis	Nauja priesaga	Algoritmas
1	„O_E(“	„)“	„“	„“	OPES
2	„P_E(“	„)“	„“	„“	Pailerio
3	„SUM_E(“	„)“	„SUM_EK(“	„) “	Pagalbinis
4	„AVG_E(“	„)“	„AVG_EK(“	„) “	Pagalbinis

lent. 2 Užklausų keitikliai ir jų parametrai

Atlikus užklausos modifikacijas su visais užklausų keitikliais, modifikuota užklausa perduodama į RDBVS naudojantis specialia klase, realizuojančia CDBVSSąsają. RDBVS užklausa vykdoma kaip įprasta SQL užklausa, išskyrus tuos atvejus, kai užklausoje yra naudojamos agregatinės „SUM_E“ arba „AVG_E“ funkcijos, tuomet RDBVS iškviečia arba atitinkamas SQL vartotojo funkcijas arba išorines vartotojo funkcijas.

RDBVS įvykdžius užklausą, rezultatai yra grąžinami į CKeitiklį ir pradedamas jų modifikavimas paeiliui perduodant juos kiekvienam rezultatų keitikliui (CRezultatųKeitiklis). Rezultatų keitikliai modifikuoja (iššifruoja) tik reikšmes tų laukų, kurie savo pavadinime turi tam tikras priesagas. Reikalingi rezultatų keitikliai ir jų parametrai pateikti žemiau (žr. lent. 3).

Nr.	Priesaga	Algoritmas
1	„_o“	OPES
2	„_p“	Pailerio

lent. 3 Rezultatų keitikliai ir jų parametrai

Rezultatų keitikliams baigus modifikuoti rezultatus, jie yra perduodami atgal į klientinę programą (programos taikomąją dalį).

7. Agregatinės funkcijos

Pagal šį sudarytą modelį RDBVS pusėje turi būti realizuotos 2 agregatinės funkcijos: SUM_E ir AVG_E. Abi šios funkcijos privalo priimti po 2 parametrus: agreguojamas reikšmes ir Pailerio viešojo rakto n dalį. Šias funkcijas galima realizuoti per SQL vartotojo funkcijas (angl. stored functions) arba per išorines vartotojo funkcijas.

Pseudokodas SUM_E funkcijai:

```
agregatinė funkcija SUM_E (duomenys, n)
{
            rezultatas = 1;
            VISIEMS x IŠ duomenys
            {
                        rezultatas = rezultatas * x mod (n^2);
            }
            GRĄŽINTI rezultatas;
}
```

Pseudokodas AVG_E funkcijai:

```
agregatinė funkcija AVG_E (duomenys, n)
{
            rezultatas = 1;
            VISIEMS x IŠ duomenys
            {
                        rezultatas = rezultatas * x mod ( n ^ 2);
            }
            rezultatas = „#“ + Kiekis(duomenys) + „#“+ rezultatas;
            GRĄŽINTI rezultatas;
}
```

AVG_E atveju dalis skaičiavimų tenka ir klientui ir RDBVS. Taip yra todėl, kad šifravimui naudojant Pailerio kriptosistemą įvesties ir išvesties duomenimis gali būti tik natūralieji skaičiai, todėl Pailerio kriptosistemos atviro teksto sandaugos homomorfinė savybė yra nenaudinga skaičiuojant vidurkį – tenka atlikti sumavimą RDBVS pusėje ir apskaičiuoti (padalinti iš dėmenų skaičius) kliento pusėje. Norint perduoti dėmenų skaičių klientui, jį galima įtraukti į perduodamą sumą (simbolių seką), pvz., atskiriant skirtukais.

IV. Daugiašalio saugumo modelio, pagrįsto šifravimu, įvertinimas

1. Modelio privalumai

Duomenys apsaugoti nuo RDBVS. Duomenys RDBVS saugomi šifruotoje formoje pačiai RDBVS neturint iššifravimo raktų, todėl net susikompromitavus RDBVS duomenys lieka saugūs.

Operacijų vykdymas RDBVS pusėje. Paieškos, sumavimo ir vidurkio operacijos vykdomos RDBVS serveryje, klientinei programai paliekant tik šifravimo ir iššifravimo veiksmus, bei pačių duomenų panaudojimą. Tai ypač svarbi savybė, kurios nebūvimas reikštų didelių duomenų srautus tarp RDVS ir kliento siekiant atlikti paiešką ir skaičiavimus kliento pusėje.

RDBVS kodo keitimo nereikalingumas. Modelis gali būti taikomas uždaro kodo RDBVS, kadangi visos reikalingos RDBVS modifikacijos atliekamos arba per SQL vartotojo funkcijas arba per išorines vartotojo funkcijas. Visos lent. 1 (žr. psl. 9) pateiktos RDBVS tokias galimybes turi.

Greita duomenų paieška. OPES algoritmas privalumas yra tas, kad duomenų reikšmės yra pakeičiamos kitomis padidinant reikalingos vietos poreikį tik nežymiai. Kitaip sakant, skaičiai yra keičiami sąlyginai panašaus dydžio (duomenų tipo prasme) skaičiais. Kadangi slaptaraštis irgi yra skaičius tokio tipo, kurį RDBVS gali nesunkiai palyginti (naudojant <, >, =, >=, <=), paieškos operacijos yra vykdomos greitai.

2. Modelio trūkumai

Duomenų dubliavimas. Tie patys duomenys yra šifruojami du kartus (dėl skirtingų šifravimo algoritmų savybių) ir jų abu slaptaraščiai yra saugomi toje pačioje reliacinėje lentelėje, tačiau skirtinguose laukuose. Informacijos apimties prasme šis dviejų laukų naudojimas tų pačių duomenų saugojimui yra resursų švaistymas: papildomų resursų sunaudojimas neįrašant į duomenų bazę naujos informacijos. Žinoma, šis dubliavimas yra būtinas, jei norima taikyti ir paiešką ir sumos, bei vidurkio agregatines funkcijas, tačiau tuo atveju, jei reikalinga tik paieška arba tik agregatinės funkcijos, duomenų dubliavimo galima atsisakyti.

Didelis resursų poreikis. Reikalingas Pailerio lauko dydis tiesiogiai priklauso nuo naudojamo Pailerio rakto ilgio. Šiuo metu saugiu yra laikomas 1024 arba ilgesnis raktas, todėl šifruojant 32 bitų reikšmes prireiks 34 kartus (64 bitai OPES laukui + 1024 bitai Pailerio laukui) daugiau vietos duomenų bazėje, nei norint saugoti nešifruotas reikšmes. Atsisakius Pailerio lauko (sumos ir vidurkio agregatinių funkcijų), vietos sąnaudos tipiniu atveju nebus didelės arba net lygiai tokios pat, kaip kad saugant nešifruotas reikšmes.

Užklausų neskaidrumas. Keitiklis keičia (iššifruoja) rezultatus pagal laukų pavadinime esančias priesagas, o užklausas modifikuoja pagal jose rastą tekstą, pvz., „O_E(15)“ bus pakeista į OPES algoritmu šifruotą reikšmę. Toks keitiklio veikimo principas reiškia, kad klientinė programa turi formuoti užklausas atsižvelgdama į tai, jog naudojamasi šifravimu. Pasiūlytas keitiklio mechanizmas nėra tinkamas tai atvejais, kai negalima modifikuoti klientinės dalies ir norima, kad užklausos ir rezultatai būtų modifikuojami nematomai.

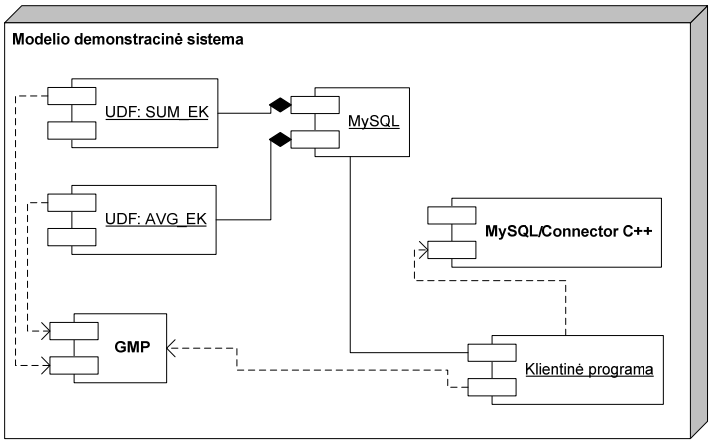
Dalinis duomenų slaptumas. OPES algoritmas šifruojant duomenis paslepia jų tikrąsias reikšmes ir tikrąjį duomenų skirstinį, tačiau išlaiko duomenų eiliškumą; to pakanka ne tik paieškos užklausų vykdymui, bet ir tam tikros informacijos išgavimui. Tarkime duomenų bazėje saugoma informacija apie skolas. Nors tikrųjų skolų didžių sužinoti neiššifruojant negalima, tačiau dėl eiliškumo buvimo

galima sužinoti, kas turi didesnę ar mažesnę skolą – kai kuriais atvejais, tai gali būti laikoma svarbia neapsaugota informacija.

Tik skaitinių duomenų apsauga. Pasiūlytas modelis tinkamas tik skaitinių duomenų apsaugai ir jo negalima pritaikyti pvz., tekstinių duomenų šifravimui.

3. Demonstracinė programa

Pasiūlytam modeliui pademonstruoti buvo sukurta speciali demonstracinė sistema, kurią sudaro komponentai, pateikti diagramoje žemiau (žr. pav. 16). Tikslios (svetimų) komponentų versijos ir naudoti įrankiai pateikti lent. 4.



pav. 16 Demonstracinė sistema

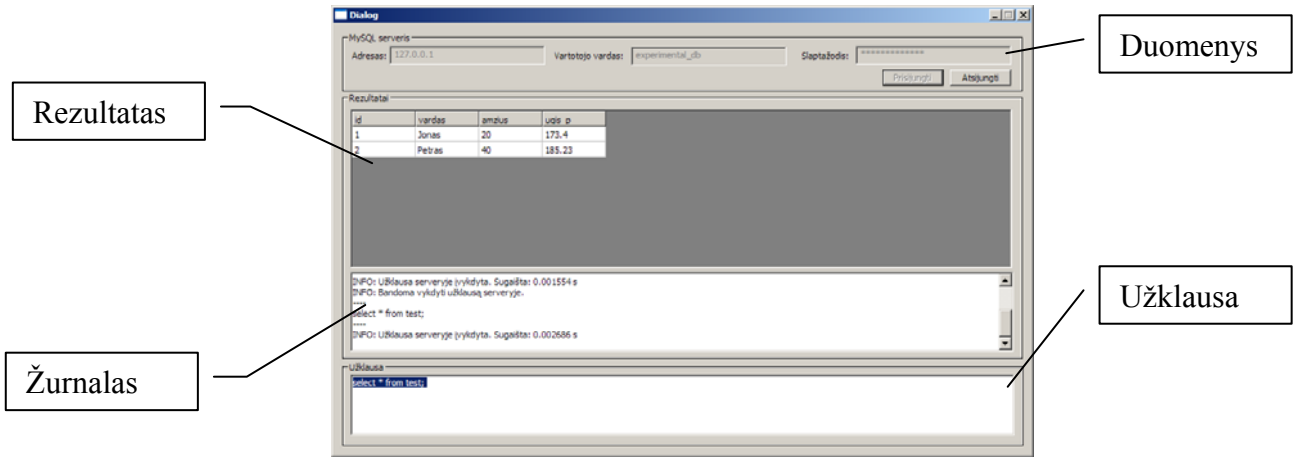
Komponentas	Versija	Aprašymas
MySQL	5.1.33	RDBVS
MySQL/Connector C++	1.0.5	C++ biblioteka, skirta duomenų mainams su MySQL
GMP	4.2.4	C biblioteka darbui su dideliais skaičiais
libpaillier	0.7	Minimalistinė C biblioteka, realizuojanti Pailerio algoritmą
MFC	9.0.30729	Biblioteka, leidžianti naudoti Windows API objektiškai
Microsoft Visual Studio	2008 (v9)	Programavimo aplinka ir C/C++ kompiliatorius
MFC Grid Control	2.26	MFC klasė – lentelė.

lent. 4 Komponentai ir įrankiai

SUM_E ir AVG_E funkcijos realizuotos kaip MySQL UDF (angl. user defined functions) – C++ kalba rašyti MySQL serverio moduliai (bibliotekos), kuriuos įkrauti ir pašalinti galima net veikiant serveriui.

Demonstracinėje sistemoje Pailerio laukams skirtos reikšmės koduojamos 1024 bitų ilgio raktu ir saugomos duomenų bazėje kaip VARCHAR. Didelių skaičių operacijos atliekamos naudojantis GMP biblioteka .

Klientinė programa priima vartotojo įvedamas užklausas, jas apdoroja (per užklausų keitiklius), perduoda į MySQL RDBVS serverį (naudojant MySQL/Connector C++) ir gautus rezultatus modifikavusi (per rezultatų keitiklius) juos atvaizduoja vartotojui kaip lentelę. Vartotojo sąsaja pavaizduota pav. 17.



pav. 17 Vartotojo sąsaja

Demonstracinės sistemoje rašant SQL užklausas, kuriomis kreipiamasi į šifruotus laukus, galima (reikia) naudoti šias funkcijas:

1. P_E(x) – užšifruoti reikšmę x Pailerio algoritmu;
2. O_E(x) – užšifruoti reikšmę x OPES algoritmu;
3. SUM_E(c) – apskaičiuoti stulpelio c sumą, kai c šifruotas Pailerio algoritmu;
4. AVG_E(c) - apskaičiuoti stulpelio c vidurkį, kai c šifruotas Pailerio algoritmu.

Norint, kad šifruotos reikšmės būtų automatiškai iššifruojamos, būtina formuojamoje užklausoje nurodyti grąžinamų šifruotų laukų pavadinimus remiantis šiomis taisyklėmis:

1. Stulpeliams, šifruotiems Pailerio algoritmu, pavadinimo gale reikia pridėti „_p“;
2. Stulpeliams, šifruotiems OPES algoritmu, pavadinimo gale reikia pridėti „_o“.

Atliekant demonstracinės sistemos testavimą buvo formuojamos užklausos, atitinkančios užklausas, pateiktas priede nr.1 . Gauti užklausų rezultatai buvo teisingi, todėl galima teigti, jog demonstracinė sistema ir saugumo modelis, kuriuo ji paremta, veikia korektiškai.

Išvados

1. Suformuluoti pagrindiniai reikalavimai, keliami daugiašalio reliacinių duomenų bazių saugumo modeliui, atskiriantys jį nuo vienašalio saugumo modelio:
    a. RDBVS turi saugoti informaciją;
    b. RDBVS turi atlikti operacijas su informacija;
    c. RDBVS negali „suvokti“ informacijos.
2. Daugiašalio reliacinių duomenų bazių saugumo modeliui keliami reikalavimai, kuomet duomenų apsaugai taikomas šifravimas, gali būti suvesti į vieną: RDBVS turi atlikti operacijas su duomenimis jų neiššifruodama.
3. Pasiūlytas daugiašalio saugumo modelis, pagrįstas šifravimu naudojant OPES ir Pailerio algoritmus.
4. Pasiūlytas modelis įvertintas analitiškai:
    a. Privalumai:
        i. RDBVS negali iššifruoti duomenų;
        ii. RDBVS gali atlikti duomenų paiešką, rasti jų sumą, bei vidurkį;
        iii. Modelis tinka uždaro kodo RDBVS, kadangi reikalingos modifikacijos, gali būti atliktos naudojant SQL vartotojo funkcijas arba vartotojo modulius;
        iv. Duomenų paieška yra greita, kadangi OPES algoritmu užšifravus reikšmę duomenų tipas nepakinta.
    b. Trūkumai:
        i. Ta pati informacija saugoma dviejuose laukuose;
        ii. Didelis resursų poreikis, atsirandantis dėl Pailerio lauko ilgio;
        iii. Galimybė RDBVS atlikti paiešką gali būti ne tik privalumu, bet ir trūkumu, kadangi reikšmių lyginimas yra dalinis jų suvokimas;
        iv. Užklausų neskaidrumas; būtinybė modeliui pritaikyti klientą;
        v. Modelis tinkamas tik skaitinių duomenų apsaugai.
5. Sukurta demonstracinė programa, įrodanti pasiūlyto modelio veikimą.

Literatūros sąrašas

1. **Silberschatz, A., Korth, H. F. ir Sudershan, S.** *Database System Concepts.* 4th. s.l. : McGraw-Hill Higher Education, 2001. psl. 1088. 0072283637.

2. **Teorey, Toby J.** *Database Modeling and Design: Logical Design.* 4th Edition. s.l. : Morgan Kaufmann, 2005. psl. 296. 0126853525.

3. *Derivability, Redundancy and Consistency of Relations Stored in Large Data Banks.* **Codd, E. F.** San Jose, California : s.n., 1969 m., IBM Research Report, T. RJ599.

4. *A relational model of data for large shared data banks.* **Codd, E. F.** 6, New York, NY, USA : ACM, 1970 m., Commun. ACM, T. 13, psl. 377-387. 0001-0782.

5. *The relational model for database management: version 2.* **Codd, E. F.** Boston, MA, USA : Addison-Wesley Longman Publishing Co., Inc., 1990 m. 0-201-14192-2.

6. **Ramakrishnan, Raghu ir Gehrke, Johannes.** *Data base management systems.* 2nd. New York : McGraw-Hill, Inc., 1999. psl. 936. 0072322063.

7. **Mata-Toledo, Ramon A. ir Cushman, Pauline K.** *Schaum's Outline of Fundamentals of Relational Databases.* s.l. : McGraw-Hill Professional, 2000. psl. 249. 007136188X.

8. Gartner Says Worldwide Relational Database Market Increased 14 Percent in 2006. *Gartner Web Site.* [Tinkle] 2007 m. 06 13 d. [Cituota: 2009 m. 03 20 d.] http://www.gartner.com/it/page.jsp?id=507466.

9. *The role of cryptography in database security.* **Maurer, Ueli.** Paris, France : ACM, 2004. Proceedings of the 2004 ACM SIGMOD international conference on Management of data. psl. 5-10. 1-58113-859-8.

10. **Natan, Ron Ben.** *Implementing Database Security and Auditing.* s.l. : Digital Press, 2005. psl. 432. 1555583342.

11. *Analysis of three multilevel security architectures.* **Levin, Timothy E., et al.** New York, NY, USA : ACM, 2007. CSAW '07: Proceedings of the 2007 ACM workshop on Computer security architecture. psl. 37-46. 978-1-59593-890-9.

12. *An introduction to multilevel secure relational database management systems.* **Rjaibi, Walid.** Markham, Ontario, Canada : IBM Press, 2004. CASCON '04: Proceedings of the 2004 conference of the Centre for Advanced Studies on Collaborative research. psl. 232-241.

13. *Hippocratic databases.* **Agrawal, Rakesh, et al.** Hong Kong, China : VLDB Endowment, 2002. VLDB '02: Proceedings of the 28th international conference on Very Large Data Bases. psl. 143-154.

14. *Privacy in data systems.* **Agrawal, Rakesh.** San Diego, California : ACM, 2003. PODS '03: Proceedings of the twenty-second ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems. psl. 37. 1-58113-670-6.

15. *On the correctness criteria of fine-grained access control in relational databases.* **Wang, Qihua, et al.** Vienna, Austria : VLDB Endowment, 2007. VLDB '07: Proceedings of the 33rd international conference on Very large data bases. psl. 555-566. 978-1-59593-649-3.

16. *Purpose based access control for privacy protection in relational database systems.* **Byun, Ji-Won ir Li, Ninghui.** 4, Secaucus, NJ, USA : Springer-Verlag New York, Inc., 2008 m., The VLDB Journal, T. 17, psl. 603-619. 1066-8888.

17. *Executing SQL over encrypted data in the database-service-provider model.* **Hacigumus, Hakan, et al.** Madison, Wisconsin : ACM, 2002. Proceedings of the 2002 ACM SIGMOD international conference on Management of data. psl. 216-227. 1-58113-497-5.

18. *Order preserving encryption for numeric data.* **Agrawal, Rakesh, et al.** Paris : ACM, 2004. Proceedings of the 2004 ACM SIGMOD international conference on Management of data. psl. 563-574. 1-58113-859-8.

19. *Answering aggregation queries in a secure system model.* **Ge, Tingjian ir Zdonik, Stan.** Viena : VLDB Endowment, 2007. Proceedings of the 33rd international conference on Very large data bases. psl. 519-530. 978-1-59593-649-3.

20. *Public-Key Cryptosystems Based on Composite Degree Residuosity.* **Paillier, Pascal.** s.l. : Springer Berlin / Heidelberg, 1999. Advances in Cryptology — EUROCRYPT '99. T. 1592, psl. 223-238. 978-3-540-65889-4.

Anotacija

Reliacinių duomenų bazių saugumo modelio tyrimas

Žilvino Brobliausko magistro studijų baigiamajame darbe atliekamas daugiašalio reliacinių duomenų bazių saugumo modelio teorinis tyrimas: suformuluojami pagrindiniai reikalavimai, keliami tokio tipo modeliui; pasiūlomas modelis, leidžiantis vykdyti paiešką ir taikyti sumos, bei vidurkio agregatines funkcijas neiššifruojant skaitinių duomenų RDBVS pusėje; nurodomi pateikto modelio privalumai ir trūkumai. Pateikiama demonstracinė programa, realizuojanti pasiūlytą modelį.

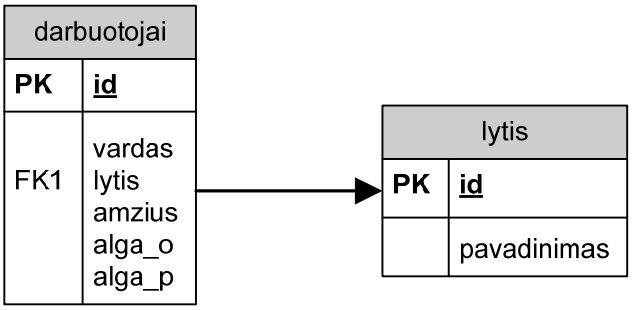
Summary

The research on security model of relational databases

The multilateral security model of relational databases is analyzed in master thesis of Žilvinas Brobliauskas. The results of research includes: the formulated requirements for multilateral security model of relational databases, proposed model, which allows range queries and aggregation functions over encrypted data without decrypting them at RDBMS level, and determined advantages and disadvantages of it. The program which realizes proposed model is given as proof of concept.

Priedas nr.1

Daugiašalio saugumo modelio, pagrįsto duomenų šifravimu, testinės SQL užklausos



pav. 18 Testinė RDB

Užklausos

Grąžinti visą saugomą informaciją:

```
select vardas, pavadinimas as lytis, amzius, alga_p from
darbuotojai, lytis where lytis=lytis.id;
```

Grąžinti visą saugomą informaciją ir neiškoduoti algos (alga_p as alga - _p nebuvimas):

```
select vardas, pavadinimas as lytis, amzius, alga_p as alga from
darbuotojai, lytis where lytis=lytis.id;
```

Grąžinti visą saugomą informaciją ir neiškoduoti algos (alga_o as alga - _o nebuvimas):

```
select vardas, pavadinimas as lytis, amzius, alga_o as alga from
darbuotojai, lytis where lytis=lytis.id;
```

Apskaičiuoti bendrą uždirbamų pinigų sumą:

```
select sum_e(alga_p) as algos_suma_p from darbuotojai;
```

Apskaičiuoti vidutinį uždarbį:

```
select avg_e(alga_p) as vidutine_alga_p from darbuotojai;
```

Apskaičiuoti vidutinį uždarbį pagal lytis:

```
select pavadinimas as lytis, avg_e(alga_p) as vidutine_alga_p from
darbuotojai, lytis where lytis=lytis.id group by lytis;
```

Sukurti naują įrašą (reikšmė šifruojamam laukui perduodama su funkcija P_E() ir O_E()):

```
insert into darbuotojai values
(null,"Martynas",1,21,P_E(1205),O_E(1205));
```

Išrikiuoti sąrašą algų didėjimo tvarka:

```
select vardas, pavadinimas as lytis, amzius, alga_p from
darbuotojai, lytis where darbuotojai.lytis=lytis.id order by
alga_o asc;
```

Surasti visus įrašus su alga = 800:

```
select vardas, pavadinimas as lytis, amzius, alga_p from
darbuotojai, lytis where darbuotojai.lytis=lytis.id and alga_o =
O_E(800);
```

Surasti visus įrašus su alga <1000:

```
select vardas, pavadinimas as lytis, amzius, alga_p from
darbuotojai, lytis where darbuotojai.lytis=lytis.id and alga_o <
O_E(1000);
```

Surasti visus įrašus, kai 800 < alga <1200:

```
select vardas, pavadinimas as lytis, amzius, alga_p from
darbuotojai, lytis where darbuotojai.lytis=lytis.id and alga_o >
O_E(800) and alga_o < O_E(1200);
```

Surasti įrašus su didžiausia alga:

```
select vardas, pavadinimas as lytis, amzius, alga_o from
darbuotojai, lytis where darbuotojai.lytis=lytis.id and alga_o in
(select max(alga_o) from darbuotojai);
```

Surasti įrašus su mažiausia alga:

```
select vardas, pavadinimas as lytis, amzius, alga_o from
darbuotojai, lytis where darbuotojai.lytis=lytis.id and alga_o in
(select min(alga_o) from darbuotojai);
```