



**Kauno technologijos universitetas**

Informatikos fakultetas

**Eligijus Kiudys**

# **Rolėmis grįstas žiniatinklio programų prieigos valdymo metodas**

Analizė

Vertino:

prof. Algimantas Venčkauskas

**Kaunas 2022**

# Turinys

Ižanga.....	5
Tikslas ir uždaviniai .....	5
Dokumento struktūra .....	5
Žiniatinklio programų prieigos valdymo problemos .....	6
Prieigos valdymas (Separation of Duty) .....	7
Dinamiškumas.....	7
Vientisumo problema tarp sistemų .....	7
Modelio valdymas.....	7
Modelio valdymo problema (Role explosion), .....	8
User Authorization Query .....	8
Sudėtingumas .....	8
Momentinė prieiga prie informacijos.....	8
Žiniatinklio programų prieigos valdymo metodai .....	9
Rolėmis grįstas prieigos valdymo metodas.....	10
RBAC Administravimas .....	12
Rolėmis grįsto prieigos valdymo metodo pritaikymas žiniatinklyje .....	13
Kitos debesų paslaugos .....	15
Projekto konceptas ir metodo konceptas.....	16
Prieigos valdymo metodai.....	19
Rolėmis grįstas prieigos valdymo metodas.....	19
Atributais grįstas prieigos valdymo metodas .....	20
Atributais-Rolėmis grįstas prieigos valdymo metodas .....	21
Technologijos.....	23
Išvados .....	24
Literatūros sąrašas.....	25

## Paveikslėlių sąrašas

pav. 1 prieigos valdymo modelio problemos .....	6
pav. 2 Rolių valdymo modelis .....	10
pav. 3 KTU Sistema. ....	13
pav. 4 Microsoft .....	13
pav. 5 Google .....	14
pav. 6 Facebook .....	14
pav. 7 Debesų paslaugos .....	15
Pav. 8 Prototipas. ....	16
Pav. 9 Duomenų srauto diagrama .....	17
Pav. 10 Use Case diagrama.....	18
Pav. 11 Rolėmis paremto valdymo metodo modelis. ....	20
Pav. 12 Atributais paremto valdymo metodo modelis .....	21
Pav. 13 Atributais-rolėmis paremto metodo modelis .....	22
Pav. 14 Rolių lentelės pavyzdys .....	22
Pav. 15 Atributų lentelės pavyzdys.....	23

## Ataskaitos pakeitimai

Sukurtas vizijos punktas apie projekto konceptą, projekto struktūrą, prieigos valdymo metodo koncepto aprašas. Pavyzdinio projekto aprašas su „Use Case“ diagrama, „Data flow“ diagrama. Atributais-rolėmis grįstas prieigos valdymo metodas – aprašas ir sudarytas konceptas.

# **Ižanga**

Rolėmis grįstas prieigos valdymo metodas yra vienas iš populiariausių metodų esančių, kurie skirti valdyti internetinių puslapių arba internetinių aplikacijų prieigą. Žinoma yra ne vienas metodas skirtas prieigos valdymui, bet dažniausiai tai būna minėto metodo variacijos arba visiškai skirtingas metodas. Prieigos valdymo problema išlieka ir šiai dienai. Vis daugiau ir daugiau plečiantis internetui ir didėjant tiekiamų paslaugų kiekiui internete reikia ir saugesnių ir patogesnių metodų valdyti internetines aplikacijas, puslapius. Rolėmis grįstas žiniatinklio programų prieigos valdymo metodas leis panaudoti rolėmis grįstą prieigos metodą žiniatinklio programoms. Kadangi žiniatinklio programų kiekis didėja reikia ir metodo, kuris leistų daugiau valdyti naudotojo prieigą prie programos funkcijų ir domenų. Metodas kurį analizuojau yra pagrindinis metodas prieigos valdymui. Analizės metu yra analizuojamos prieigos valdymo problemos, išsiaiškinama apie kitus esamus metodus.

## **Tikslas ir uždaviniai**

Darbo tikslas – sukurti rolėmis grįstą prieigos valdymo metodą skirtą internetinėms programoms. Metodas turėtų leisti pritaikyti rolėmis grįstą prieigos valdymo metodą internetinių aplikacijų prieigos valdymui. Panaudojus naują metodą administratorius galės keisti roles, licencijas, naujoms internetinėms aplikacijoms.

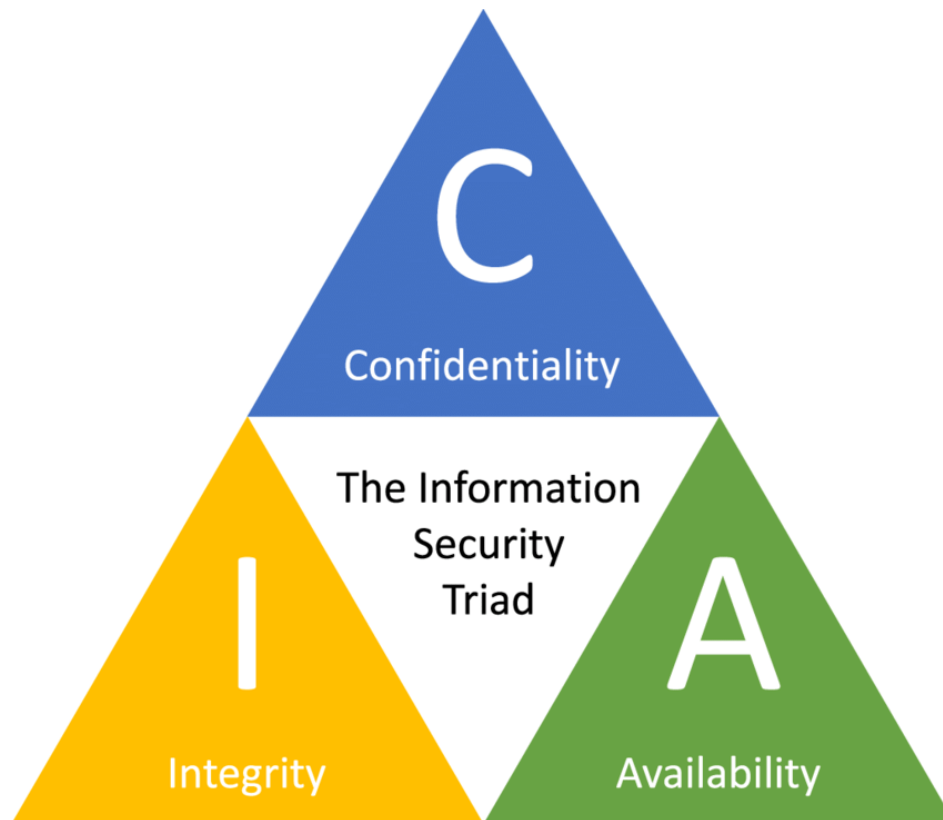
- Išanalizuoti žiniatinklio programų prieigos valdymo problemas.
- Išanalizuoti žiniatinklio programų prieigos valdymo metodus.
- Išanalizuoti rolėmis grįstas prieigos valdymo metodus.
- Išanalizuoti rolėmis grįsto prieigos valdymo metodo pritaikymą žiniatinklyje.
- Pasiūlyti rolėmis grįstą prieigos valdymo metodą skirtą žiniatinklių programoms.
- Įgyvendinti sukurtą metodą ir palyginti rezultatus.

## **Dokumento struktūra**

Darbą sudaro keturi pagrindiniai skyriai - žiniatinklio programų prieigos valdymo problemos, žiniatinklio programų prieigos valdymo metodus, rolėmis grįstas prieigos valdymo metodas, rolėmis grįsto prieigos valdymo metodo žiniatinklyje modelį analizės skyriai. Pirmame skyriuje yra analizuojamos žiniatinklio programų prieigos valdymo esamos problemos. Kitame skyriuje yra analizuojama skirtingi metodai skirti valdyti žiniatinklio programų prieigą. Trečiame skyriuje yra analizuojamas rolėmis grįsto prieigos valdymo metodą, kaip veikia minimas metodas ir šiek tiek istorijos apie metodą. Ketvirtame skyriuje yra analizuojami rolėmis grįsto prieigos valdymo metodo žiniatinklyje.

## Žiniatinklio programų prieigos valdymo problemos

Prieigos valdymo problema egzistuoja ne vienerius metus. Ar tai būtų internete ar bibliotekoje, ar universitete. Visą laiką reikia verifikuoti, kad esi tikrai tu, vienokiu ar kitokių būdu. Realybėje yra naudojamos ID kortelės arba pasas. Realybėje galima pamatyti paso nuotrauką ir žmogaus veidą, taip verifikuojant, kad jis tikrai tas žmogus. Internetinėse aplikacijose būtų lengva verifikuoti tapatybę su ID kortele arba pasu, bet juos perkelti į elektroninę erdvę nėra labai lengva. Internetinėse aplikacijos neužtenka patvirtinti naudotojo tapatybę, bet reikia valdyti naudotojo prieigą vienu ar kitu būdu. Problemai spręsti visos internetinės aplikacijos ir net tik naudoja pasirinktą modelį naudotojų prieigos valdymui. Kiekvienas modelis turi savus plusus ir minusus. Sukurti ar panaudoti esamą prieigos valdymo modelį yra sunki užduotis, kuri kankina interneto aplikacijų kūrėjus ne vienerius metus. Naudojamo prieigos modelio užduotis yra paprasta, prileisti naudotojus prie jiems galimos informacijos, nei daugiau, nei mažiau.



*pav. 1 prieigos valdymo modelio problemos*

- Konfidencialumas
- Vientisumas
- Prieinamumas

## **Prieigos valdymas (Separation of Duty)**

Viena iš pirmųjų problem yra naudotojų prieigos atsikyrimas. Šia problem bando išspręsti kiekvienas prieigos saugos modelis. Problema atsirado, kai keletas naudotojų pradėjo naudotis tokiu pačiu kompiuteriu, tada iškilo klausimas. Kaip galima atskirti naudotojo prieinamus resursus kiekvienam naudotojui, pavyzdžiui, kokias aplikacijas gali naudoti naudotojas, o kokių negali. arba kokius failus gali redaguoti, o kokių negali. Minėta problema išlieka ir dabar. Naudojant internetines sistemas reikia atskirti teikiamus resursus pagal naudotojo prieigą.

## **Dinamiškumas**

Pradėkime nuo pirmos problemos: modelio dinamiškumo. Žinoma pasirinkus prieigos valdymo modelį pradžioje, atrodo, kad gali tikt, bet kuris pasirinktas modelis. Padarius projektą ir administratoriui sukonfigūravus teises naudotojams atrodo viskas veikia kaip turėtų veikti. Pradedant plėsti esamą sistemą kūrėjai gali susidurti prieigos valdymo problemomis, kadangi ne visi modeliai leidžia dinamiškai plėsti projektą. Plečiant dinamišką sistemą su blogu prieigos valdymo metodu, dažniausiai yra pradedami naudoti įvairūs apėjimai, kad sistema veiktų. Tokios problemos sistemos administratoriams apsunkina darbą, kadangi realizacija yra neintuityvi, administratoriai valdant tokį projektą gali pridaryti klaidų, kurios veda prie kitų prieigos valdymo problemų.

## **Vientisumo problema tarp sistemų**

Prisijungimas prie sistemos per kitus puslapius. Naudojantis internetinėmis aplikacijomis galima prisijungti prie sistemos naudojant kitą sistemą. Dažnai iškyla įvairios problemos prisijungiant su kita sistema, priklausant nuo pasirinkto prieigos valdymo modelio. Viena iš pagrindinių problemų yra naudotojo teisės. Tarp naudojamų metodų turi būti sudarytas susitarimas, kaip koks prieigos valdymo metodas su kitu metodu, jei norima prisijungti prie skirtingų internetinių sistemų su viena paskyra.

## **Modelio valdymas**

Modelio valdymas yra dar viena problema su kuria susiduriame pritaikant pasirinktą prieigos valdymo metodą. Pritaikius pasirinktą prieigos valdymo metodą, administratorius dažniausiai turi valdyti naudotojų dalinę arba pilną prieigą. Dažnai neintuityvus modelio valdymas gali privesti prie administratoriaus klaidų, kurios suteikia naudotojams prie mažai teisių arba per daug. Žiūrint iš rolėmis grysto ar kitokio saugos modelio perspektyvos, visada gali atsirasti valdymo prieigos spragų. Paėmus pavyzdį kaip atributais paremta prieigos valdymas, panaudojant blogą kontekstą implementacijos metu, gali atsirasti prieigos valdymo spragos, naudotojas kuris neturi teises prieiti prie informacijos, ją gali pasiekti.

## **Modelio valdymo problema (Role explosion),**

Kiekviena sistema turi saugumo spragų, ar tai būtų pritaikymo problemos, ar tai būtų administravimo spragos, ar kažkokios kitos klaidos kurios priveda prie nesaugios sistemos. Vienas iš pavyzdžių būtų rolėmis paremto prieigos metodo rolių sprogimo problema. Šitame pavyzdyje yra aprašoma kaip rolių kiekis didėja iki tokių skaičių kai jų nebegalima suvaldyti. Pavyzdžiui yra vienas naudotojas, dešimt sistemų ir dvi rolės per sistemą. Naudotojas reikalauja dviejų rolių per aplikaciją gaunasi taip, kad reikalauja dvidešimt rolių. Turint daugiau negu vieną naudotoją ir jiems visiems prašant dviejų rolių per aplikaciją, tikėtina kad tvarkant roles administratorius gali įvelti ne vieną klaidą.

## **User Authorization Query**

Problema, kuri aprašo prieigos reikalavimo problemą, kaip galima efektyviai pareikalauti teisę į tam tikrą sistemą ar resursą. Ši problema ypač iškyla naudojant rolėmis grystą metodą. Norint gauti papildomą prieigą prie resursų reikia nuspresti kokią rolę reikia suteikti, kad būtų suteikti tam tikri resursai, ar tai būtų galima pakeisti esamas roles i kažkokią vieną rolę.

## **Sudėtingumas**

Sistemos sudėtingumas irgi yra problema. Norint integruoti pasirinktą prieigos valdymo metodą, sistemos sudėtingumas gali kišti koją. Programuotojai gali nesuprasti sudėtingos sistemos, ją pritaikant gali padaryti saugos klaidų. Žinoma administratoriui bus lengviau dirbti, bet kartai geriau yra pasirinkti sunkiau valdomą sistemą, bet lengviau suprantamą ir įdiegiamą.

## **Momentinė prieiga prie informacijos**

Paprastas prieigos valdymo saugos modelis, nesprendžia reikiamos momentinės prieigos problemos, kai naudotojui reikia prieigos prie sistemos vienam ar keliesm kartams ir po to jos nebereikia. Atsiranda problemos kaip reikėtų suteikti tokią prieigą, kad nebūtų galima prieiti prie kitų resursų. Kaip galima efektyviai suteikti naudotojui tokią galimybę ir kaip užtikrinti, kad naudotojas tikrai gali prieiti prie prašomų resursų.

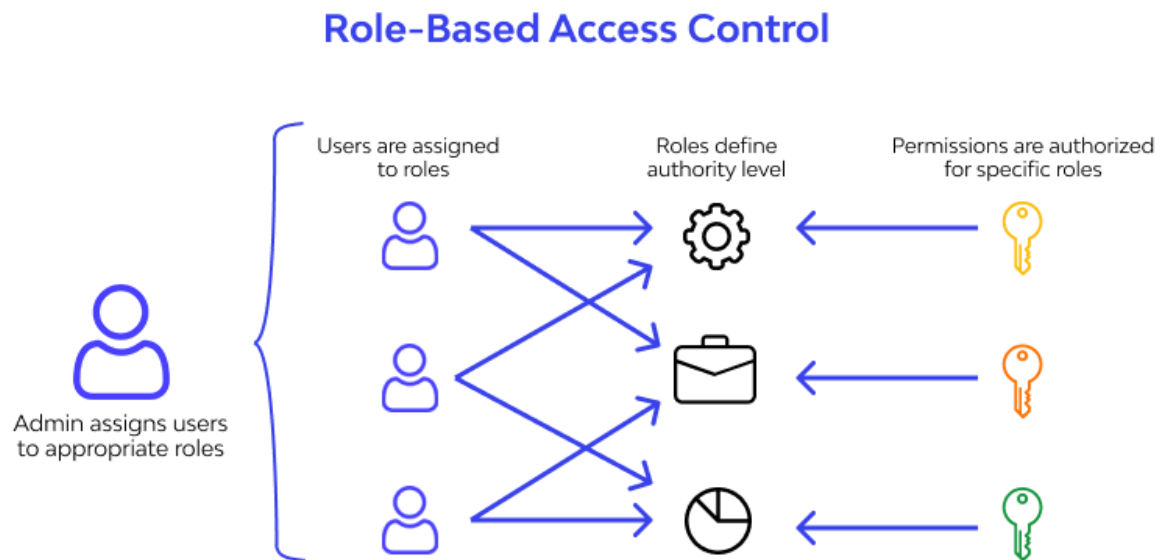


# Žiniatinklio programų prieigos valdymo metodai

- Object-Specific Role-Based Access Control (ORAC)[2]  
Konkrečių objektų rolėmis pagrįstas prieigos valdymo metodas. Šitas metodas valdo naudotojų prieigą rolėmis ir specifiniais objektais. Administratorius gali nustatyti naudotojui rolę, arba specifinį objektą, prie kurio naudotojas gali prieiti.
- Attribute-based access control[3]  
Naudojant atributais pagrįsto prieigos valdymo metodo prieiga prie išteklių gali būti nustatyta pagal skirtingus požymius pvz. vardas, IP adresas, laikas ir tt. Pagrindinė idėja ABAC yra, kad nėra tiesiogiai priskirti leidimų vartotojui, dėl to objektų prieiga leidžia visų objektų prieigą remiantis atributais. Atributas vaidina svarbų vaidmenį sistemoje ABAC leidimų suteikimas įgaliojams vartotojams, pvz., vardas, vieta, IP adresas, vieta ir tt. Atributo reikšmė nusprendžia, ar vartotojas yra įgaliojamas naudoti tam tikrą išteklių, ar ne. Vartotojas taip pat gali būti nurodyta kaip subjektas.
- Role-based Access Control[4]  
Pagrindinė RBAC koncepcija yra rolės, kurios gali parodyti prieigos kontrolės politiką tam tikrai organizacijai, institucijai ar įmonei. Leidimai sukuriama, kai objektai atliekami veiksmai, o po to šioms rolės priskiriami leidimai. Vartotojai nėra tiesiogiai priskirtas leidimams. Rolė yra tiltas tarp leidimų ir vartotojai. Vartotojams priskiriami nurodytos rolės, kad jie galėtų naudotis skirtingais leidimais.
- Attributed Role Based Access Control Model[5]  
Modelis naudoja roles kaip tiltą tarp objekto leidimų ir naudotojo. Administratorius naudotojui nustato pasirinktą rolę. Šitas modelis skiriasi nuo paprasto rolėmis pagrįsto prieigos metodo tuo, kad objektų leidimai yra priskiriami automatiškai prie rolių, naudojant atributais paremtą prieigos valdymo modelio. Rolės yra išskirstytos lygiais, pvz. naudotojas kuri priklauso pirmo lygio rolei gali atlikti skaitymo, rašymo, tvarkymo ir trynimo veiksmus.
- RBAC-SC: Role-based Access Control using Smart Contract[6]  
Rolėmis grįstas prieigos valdymo metodas naudojant išmanų kontraktą, veikia labai panašiai kaip Rolėmis grįstas prieigos valdymo metodas, tik yra vienas skirtumas, minėtas metodas naudoja blockchain technologiją saugiai gauti rolės prieigą ir atlikti tik naudotojui skirtas funkcijas.
- Intent-Based Access Control (IBAC)[7]  
Šiek tiek istorijos apie prieigos valdymo metodus. Pradedant naudoti kompiuteriais, žmonės suprato, kad reikėjo neleisti naudotojams neleisti vienas kitam kištis į darbus, kai naudotojai naudojami vienu kompiuteriu. Problemai spręsti buvo sukurtas IBAC modelis, kuris priklauso nuo naudotojo tapatybės. Leidimas naudoti sistemos resursus pvz. Failus, buvo indeksuojami pagal vartotojo tapatybę, tai reiškia, kad failą gali redaguoti vienas arba keli žmonės priklausant nuo leidimo.

# Rolėmis grįstas prieigos valdymo metodas

Taigi kas yra ta rolėmis grįstas prieigos valdymo metodas. Pagrindinė RBAC koncepcija yra rolės, kurios valdo naudotojo prieigą prie sistemos. Rolės yra tiltas tarp naudotojo ir sistemos licencijų. Naudotojas gavęs rolę gali pasiekti administratoriaus skirtą funkcionalumą. Administratorius valdo naudotojus, roles, licencijas. Dažniausiai administratoriui yra skirta valdymo rolė, su kuria gali valdyti naudotojus, roles ryšiu ir t.t. Niekas kitas be administratoriaus negali prieiti prie rolių valdymo.



pav. 2 Rolių valdymo modelis

Rolėmis grįstas prieigos metodas atsirado 1990-ais, kaip patikima technologija valdant dideles sistemas. Rolėmis grįsto prieigos metodo paprastas paaiškinimas yra naudotojams yra priskirtos rolės kurios yra susietos su leidimais. Toks metodas palengvina sistemos valdymą. Galime pagalvoti, kas būtų jei nebūtu rolėmis pagrįsto prieigos valdymo metodo. Greičiausia arba naudotume senesnę modelį arba dar nematytą modelį.[7]

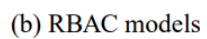
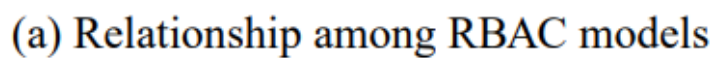


Fig. 1. A family of RBAC models.

Be abejonų, ši technologija yra dabar viena iš populiariausių technologijų, valdant prieigą internete ir įvairiuose įrenginiuose.

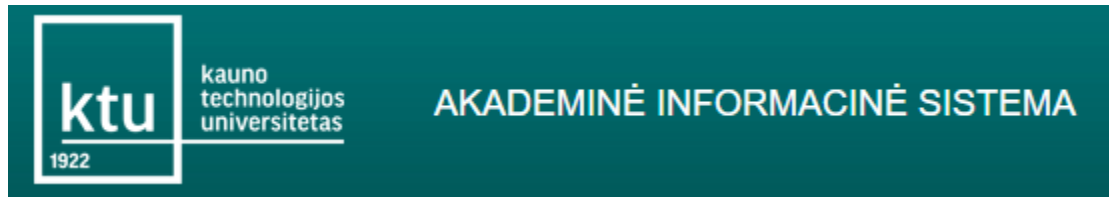
Naudodami RBAC, sistemos administratoriai gali kurti roles, suteikti toms rolėms leidimus ir priskirti vartotojų roles pagal jų konkrečias darbo pareigas ir politiką. Visų pirma, vaidmenų ir teisių ryšiai gali būti nustatyti iš anksto, todėl vartotojus lengva priskirti iš anksto nustatytoms rolėms. Be RBAC būtų sunku nustatyti, kokie leidimai turi būti suteikti naudotojams.

## **RBAC Administravimas**

Pradėkime nuo to kas valdo roles. Tai roles valdo administratorius. Administratorius yra asmuo kuris sutvarko sistemos saugos politiką. Administratorius atlieka sistemos auditą, tvarko sistemos roles, tvarko sistemos leidimus kurie yra susieti su rolėmis, kitaip sakant priskiria leidimus prie rolių. Administratorius yra pagrindinis asmuo kuris prižiūri sistemos saugumą. [8]

Į administravimo funkcijas įeina leidimų kūrimas ir palaikymas elementams. Su administravimo funkcijomis galima sukurti naudotojus, roles, operacijas ir objektus. Turint prieigą prie tokių funkcijų administratorius gali valdyti visos sistemos prieigą naudotojams ir naudotojų kiekį. Naudojant administravimo funkcijas galime nustatyti ryšius tarp naudotojų, rolių ir objektų. Pasinaudojus funkcijomis taip pat galima ir panaikinti sukurtus ryšius. Tokios funkcijos suteikia administratoriui valdyti visą sistemą efektyviai ir dinamiškai. Žinoma augant sistemai auga ir darbo kiekis, kadangi reikia valdyti didesnę kiekį rolių ir jų ryšių.

## Rolėmis grįsto prieigos valdymo metodo pritaikymas žiniatinklyje



*pav. 3 KTU Sistema.*

Galime pradėti nuo dažniausiai studentų naudojamų sistemų, kaip moodle ir KTU akademinės sistemų. Minėtos sistemos turi vieną naudojamą prisijungimą. Kiekviena sistema valdo prieigą skitą naudotojui, pagal reikiamą funkcionalumą. Galime palyginti studento ir destytojo prieigą. Studentas gali peržiūrėti, ką destytojas yra ikėlęs ir naudotis pateikta medžiaga. Destytojas gali tvarkyti ikeltą medžiagą, trinti, keisti ir pridėti, destytojas turi turėti skirtingus leidimus nei studentas, kad galėtų tvarkyti reikiamus duomenis. Naudonat KTU sistemos prisijungimus galima prisijungti ir prie kitų sistemų, kaip Microsoft sistemų, moodle sistemos ir kitų.



*pav. 4 Microsoft*

Microsof kompanija naudoja vieną iš prieigos valdymo metodų. Mircrosoft teikia ne vieną paslaugą kuri reikalauja prisijungti prie sistemos, kaip Outlook, Azure, Windows, One Drive, Office. Su kompanijos paskyra galima prisijungti prie visų minėtų sistemų. Kiekviena sistema suteikia skirtingą prieigą prisijungus prie sistemos. Vienas prisijungimas leidžia lengvai pasiekti visas teikiamas paslaugas. Visoms šioms paslaugoms reikia valdyti suteikiamus leidimus. Kiekviena sistema suteikia skirtingus leidimus. Su Microsoft paskyra taip pat galima prisijungti ir prie kitų sistemų, kurios palaiko paskyrą. Prisijungimas suteikia paskyrios duomenis kurie yra naudojami kitose sistemose.



*pav. 5 Google*

Didžioji dalis žmonių naudoja Google paslaugomis. Kompanija turi ne vieną paslaugą kaip, gmail, drive, docs ir daug kitų paslaugų. Gauti prieigą prie minėtų paslaugų yra ganėtinai lengva. Užtenga turėti Google paskyrą ir yra gaunama prieiga prie įvairių paslaugų. Žinoma paslaugos turi skirtingus leidimus. Galime išanalizuoti Google Drive paslaugą. Naudotojai kurie naudoja minėtą paslaugą nemokamai turi mažiau vietos, negu naudotojai, kurie moka mėnesinį mokestį. Čia yra tik vienas iš pavyzdžių kur yra naudojamas prieigos valdymas. Žinoma yra ir ne Google sistemų kurios naudotojo prisijungimui naudoja Google paskyras, tokių sistemų prieigos valdymas yra valdomas pačiose sistemose.

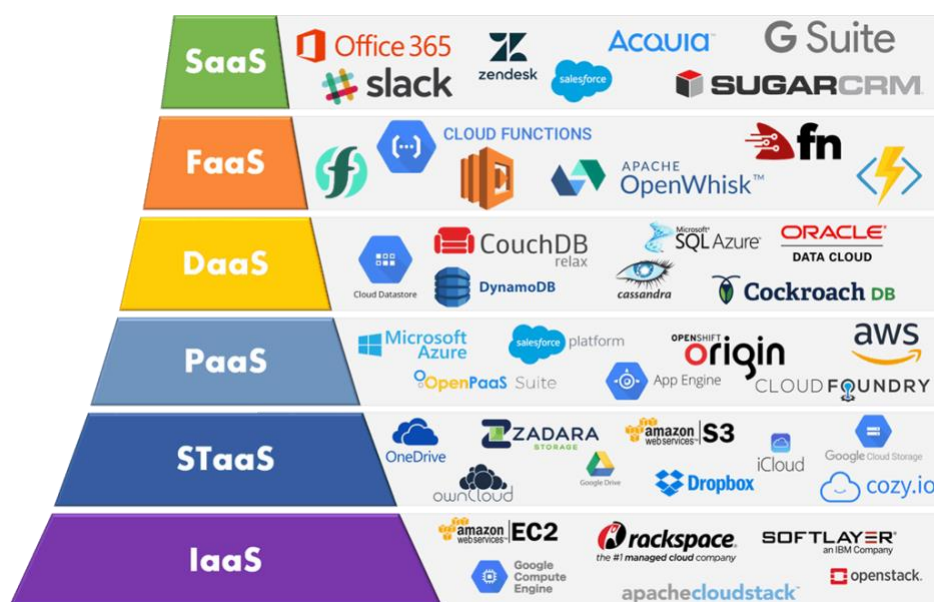


*pav. 6 Facebook*

Klausimas kyla kodėl butu galima panaudoti socialiniuose tinkluose. Užtenka truputį pagalvoti ir viskas tampa aišku. Socialinėse medijos paaiškintas prieigos valdymo metodas yra naudojamas naudotojų ir prieigos valdymui. Socialinės medijos turi didelį funkcionalumo kiekį, didelį vidinių grupių bei vidinių puslapių kiekį. Kiekvieną puslapį ar grupę kažkas valdo. Būtent tam valdymui yra naudojamas rolėmis pagrįstą prieigos valdymo metodas, arba nors panašus metodas kuris remiasi šiuo metodu. Paaiškintas naudojimas yra tik vienas iš keleto. Kaip ir Microsoft ir Google, kompanija suteikia galimybę prisijungti prie kitų sistemų su viena paskyra. Jei kitos sistemos integruoja prisijungimą su Facebook paskyra, jos valdymą sukurtoje sistemoje vistiek reikia sukurti, kadangi tokios kompanijos suteikia integraciją paskyros prisijungimui.

## Kitos debesų paslaugos

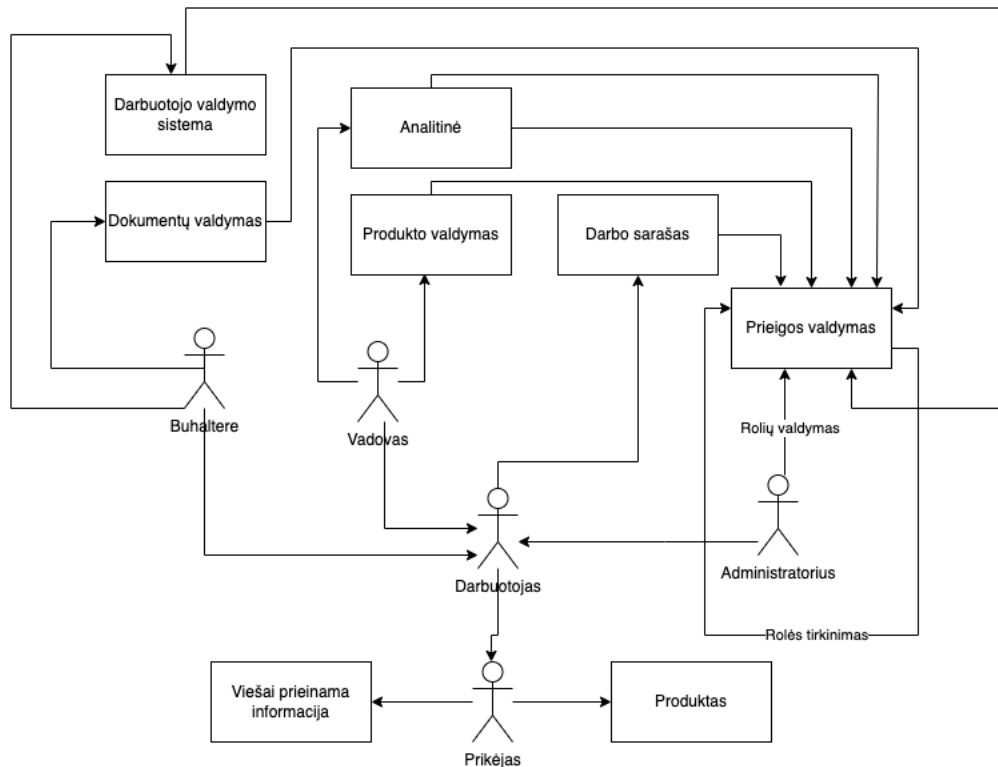
Vis dažniau ir dažniau išgirstame terminą debesų paslaugos. Taigi pirma išsiaiškinkime kas yra debesų paslaugos ir tada galėsime pradėti kalbėti apie rėlėmis grįsto prieigos metodo taikymą šiuose paslaugose. Debesų paslaugos yra teikiamos įvairios paslaugos per internetą. [10] Tokioms paslaugoms reikia valdyti naudotojų prieigą prie aplikacijos ar paslaugos. Žinoma galima įvairiai valdyti naudotojų prieigą, bet šiuo momentu kalbame apie rėlėmis grįsto prieigos metodą. Taigi kaip galima pritaikyti rėlėmis grįstą prieigos metodą debesio paslaugoms. Na pirmiausia kaip ir prie visų paslaugų naudotojas turi prisijungti prie sistemos. Paslaugų teikimas gali būti nevienodas visiems, tam galima pritaikyti rėlėmis grįstą prieigos metodą. Kiekviena rolė turi prieigą prie tam tikrų paslaugų. Žinoma administratoriui atitenka daug darbo reguliuoti roles, todėl yra naudojamas kitas metodas arba automatinis rolių valdymo metodas.



pav. 7 Debesų paslaugos

# Projekto konceptas ir metodo konceptas

Rolėmis grįstas prieigos valdymo metodas yra naudojamas didelėse sistemose, prieigos valdymui. Prieigos valdymas padeda valdyti naudotojus ir jų prieigą prie sistemų. Mano sistema nėra labai didelė, bet koncepto įrodymui užtenka ir mažos sistemos.



Pav. 8 Prototipas.

Trumpas paaiškinimas apie sistemą. Sistemos naudojimas yra įmonės viduje. Projektuota sistema yra mažo dydžio įmonės valdymo sistema. Ši sistema padeda susitvarkyti dokumentus, darbų sąrašus, atlikti darbuotojų analizę, bei valdyti pačius darbuotojus.

Naudotojo prototipe yra parodomas būsimos sistemos prototipas. Trumpai paaiškinant nubraižytą prototipą sistema yra apie įmonės vidinį valdymą. Kiekvienas komponentas yra skirtinga sistemos dalis.

Sistemos dalys ir jų aprašas:

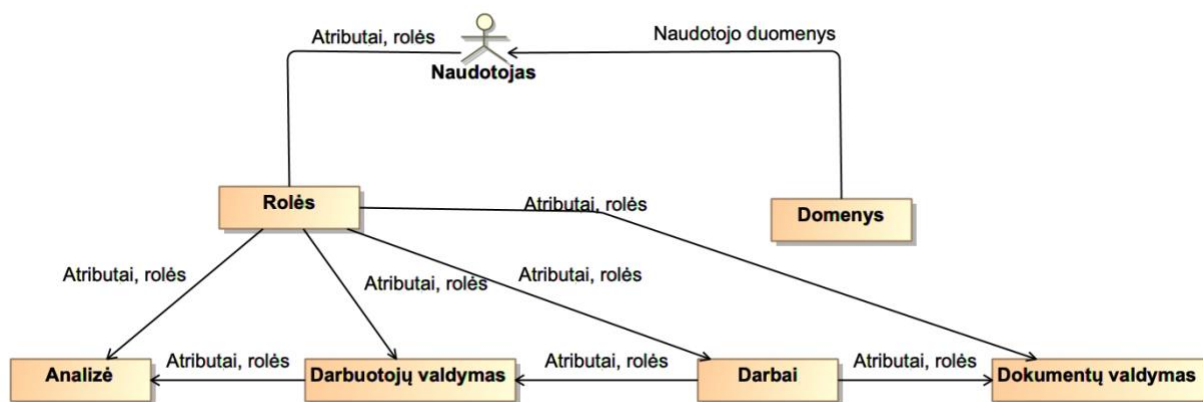
- Darbuotojo valdymo sistema – sistema skirta valdyti naudotojų duomenis, algą ir kitą informaciją
- Dokumentų valdymas – buhalterijos dokumentai
- Analitinė dalis – produkto bei darbuotojų atliktų darbų analizė
- Produkto valdymas – produkto valdymo sisteminė dalis
- Darbų sąrašas – darbų sąrašas ir jo valdymas kuris skirtas darbuotojams
- Prieigos valdymas – Rolių bei atributų prieigos prie sistemų valdymas



- Produktas – įmonės teikiamas produktas
- Viešai prieinama informacija

Duomenys, kurie galimai bus naudojami sistemoje

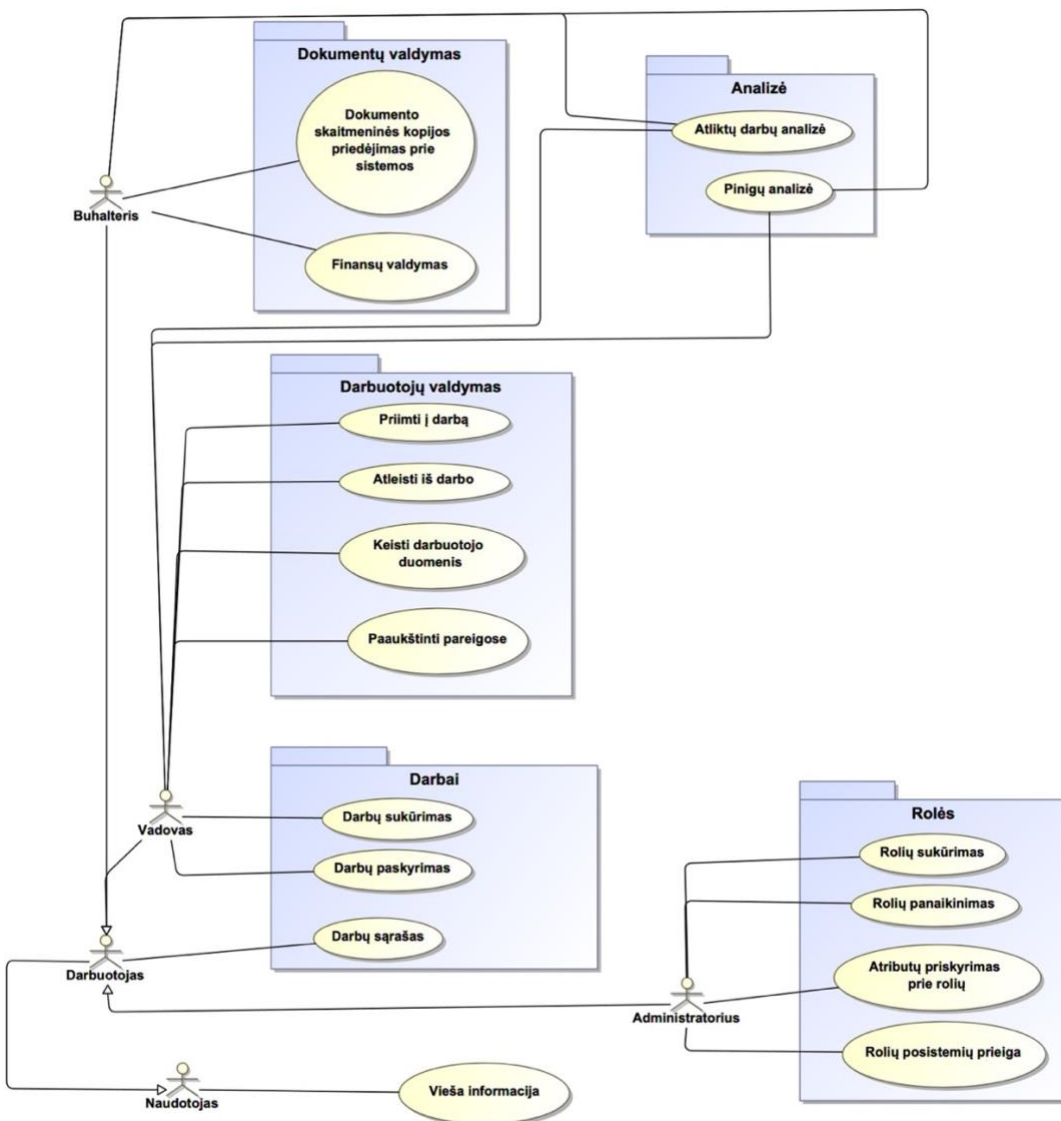
- Darbuotojų duomenys
  - Vardas
  - Pavardė
  - Adresas
  - Pozicija
  - El. paštas
  - Slaptažodis
- Atributų duomenys
  - Analizės prieigai reikalingas atributas
  - Darbų valdymas reikalingas atributas
  - Prieiga prie duomenų valdymo reikalingas atributas
- Darbų duomenys
  - Darbo aprašas
  - Darbo pavadinimas
  - Priskirtas naudotojui darbui
- Rolių duomenys
  - Sudaryta rolių matrica, prie kurių sistemų naudotojas turi prieigą
  - Naudotojas
  - Posistemės pavadinimas
  - Rolė
- Dokumentų informacija
  - Dokumentai



Pav. 9 Duomenų srauto diagrama

Sistemoje bus duomenys, kurie pastoviai bus persiunčiami, kadangi to reikalauja atributais-rolėmis grįstas prieigos metodas. Naudotojui norint pasiekti kitą posistemę reikia turėti atributus ir roles, pagal kuriuos sistema gali patikrinti, ar naudotojas gali gauti prieigą prie posistemės.

Duomenys turi būti persiunčiami, kadangi gauti visus duomenis iš duomenų bazės ar failo nėra efektyvu. Dažniausiai būna persiunčiami tokie duomenys, kaip darbuotojo duomenys, atributų duomenys ir rolių naudotojo duomenys.



Pav. 10 Use Case diagrama

Diagrama yra suskirstyta į skirtingas posistemės (mini procesus) prie kiekvieno proceso gali prieiti tik naudotojas, kuris turi reikiamą prieigą. Darbo sistema yra atsakinga už virtualią darbo aplinką. Ši aplinka yra skirta darbų priežiūrai, darbuotojų priežiūrai ir finansų valdymui. Sistemoje bus bent penki skirtingi šios sistemos naudotojai: Vadovas, Darbuotojas, Naudotojas, Administratorius, Buhalteris. Kiekvienas naudotojas turės skirtingą prieigą prie sistemos. Vadovas galės pasiekti darbų sistemą, analizės sistemą ir darbuotojų valdymo sistemą, bet prireikus jam gali būti skirtas atributas, kuris leis pasiekti daugiau posistemų. Darbuotojas galės pasiekti darbų sąrašą ir viešą informaciją, kadangi paprastam darbuotojui nieko daugiau nereikės. Administratorius, galės pasiekti rolių posistemę, tai reiškia, kad jis bus tas darbuotojas kuris bus

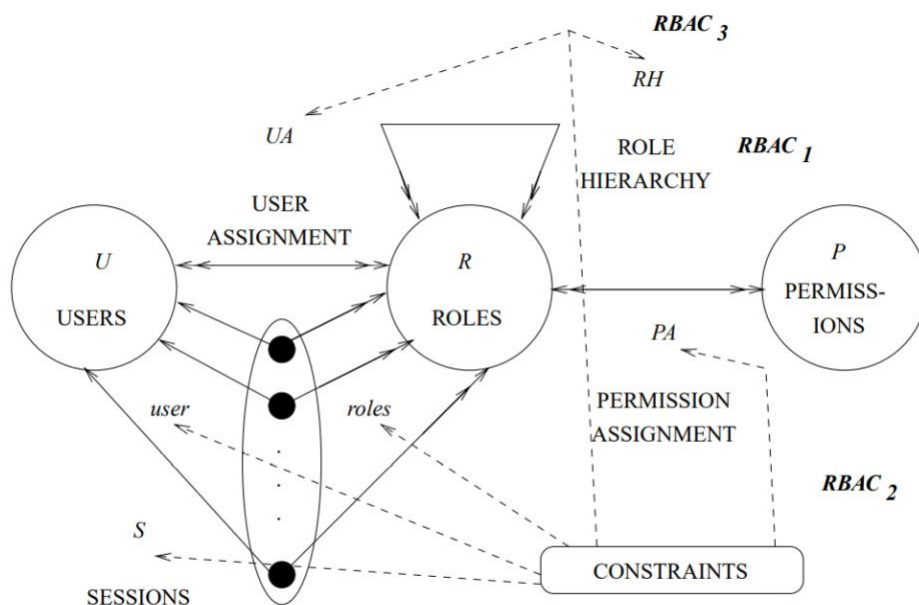
atsakingas už sistemos priežiūros valdymą. Naudotojas bus paprastas naudotojas, kuris galės gauti prieigą tik prie viešos informacijos. Buhalteris galės pasiekti analizės posistemę ir dokumentų valdymo posistemes.

## **Prieigos valdymo metodai**

Problemos prieigos valdymo methoduose yra ne viena problema, bet pati pagrindinė yra saugumas ir jo valdymas. Saugumas yra viena iš didžiausių problemų apskritai. Nesaugios sistemos yra greitai nulaužiamos. Dabar pradėkime apie Rolėmis grįsto prieigos valdymo metodo ir atributais grįsto prieigos valdymo metodo problemas, kurios iškyla naudojantis šiais metodais. Naudojantis rolėmis grįstu prieigos valdymo metodu yra sunku, kadangi rolių kiekis yra didelis, kaip ir posistemių kiekis dažniausiai būna didelis. Dažniausiai roles valdo administratorius, kuris valdant tokias sistemas gali įvelti daug klaidų, kurios gali likti nepastebėtos. Rolėmis grįsto prieigos valdymo metodo pagrindinė problema yra dinamiškumas. Kadangi rolės yra kuriamos ir priskiriamos reikia bent vieno darbuotojo, kuris jas prižiūrėtų. Yra sukurtos sistemos, kurios tai padeda išspręsti, bet jos būna labai didelės. Paėmus kitą metodą, kuris yra atributais grįstas prieigos valdymo metodas, jis taip pat nėra tobulas. Šitas metodas yra dinamiškas, bet tai reiškia, kad yra sunkus valdymas, kiekvienas posistemė turi turėti savo taisykles, kurių pagalba bus nustatyta ar naudotojo esami atributai atitinka taisykles. Naudojant paremtą prieigos valdymo metodą iškyla problemų bandant prieiti prie posistemės prieigos per visiškai kitą sistemą, kadangi šitas metodas yra statinis, todėl negalima nuspręsti, kokias roles galima skirti naudotojui iš kitos sistemos.

## **Rolėmis grįstas prieigos valdymo metodas**

Rolėmis grįstas metodas suteikia prieigą tik skirtiems asmenims. Rolės turi asociacijas su funkcijomis ar mini sistemomis. Kiekviena rolė suteikia skirtingą prieigą prie skirtingos sistemos tai leidžia atskirti naudotojų prieigą ir juos pačius, kai naudotojas naudoja sistemą. Šitas standartas yra naudojamas jau seniai. Pavyzdžiui Microsoft naudoja šitą valdymo sistemą Azure paslaugoms. Kadangi šitas metodas vis dar naudojamas yra įvairiose kompanijose jis yra patikimas, bet su savo spragomis. Pritaikymo pavyzdys mano sistemoje. Pateikiamas pavyzdys yra tik iš vienos pusės, kadangi mano sistemoje bus naudojamas hibridinis metodas.



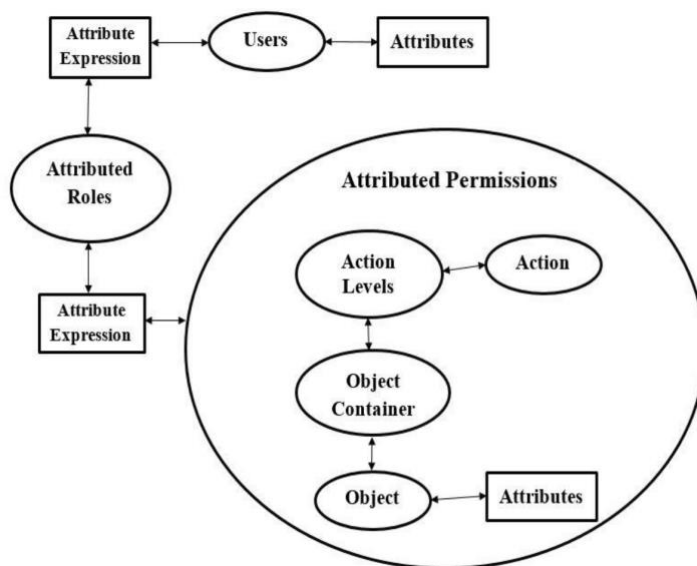
Pav. 11 Rolėmis paremta valdymo metodo modelis.

Pradėkime nuo to, kad reikia turėti registruotų naudotojų sąrašą, su naudotojų informaciją. Tada yra sukuriamas rolių asociacijų sąrašas su paslaugomis, turint tokį sąrašą galima pradėti valdyti prieigą prie paslaugų. Rolės turi būti susietos su teikiamomis paslaugomis, kitu atveju arba niekas neturės prieigos prie paslaugų, arba visi turės prieigą prie visų paslaugų. Turi būti ir paslaugų sąrašas, kitaip nebus galima priskirti rolių prie tam tikrų paslaugų. Turint tris sąrašus naudotojų, paslaugų ir rolių galima pagalvoti kur visi duomenys bus saugomi. Duomenų saugojimui pasirinkau MySQL duombazę. Tai reiškia, kad visi duomenys bus saugomi sistemos duombazėje. Toks pasirinkimas leis greitai tvarkyti reikalingus duomenis, bei leidžia greitą prieigą tikrinant roles ir sistemas. Sudarius sąrašus galima apjungti visą sistemą. Sudarius visus sąrašus viskas turi būti apjungta. Rolės su paslaugomis turi būti surištos arba atvirkščiai. Surišus roles galima jas priskirti prie naudotojų. Kiekvienas naudotojas turi turėti priskirtą rolę, kad gauti prieigą prie reikalingos darbo posistemės. Galima pradėti spręsti, kaip bus tikrinamos rolės ir kaip veiks prieiga prie posistemių. Pirmiausia bus patikrinama posistemė prie kurios bandoma prieiti. Gavus posistemės duomenis galima gauti rolių matricą, naudotojui bandant jungtis prie posistemės yra gaunama jo rolė. Gavus naudotojo rolę galima pradėti tikrinti rolę gautoje rolių matricoje. Patikrinus ar su naudotojo role galima prieiti prie posistemės naudotojas gauna arba negauna prieigą prie posistemės.

## Atributais grįstas prieigos valdymo metodas

Atributais paremtas saugos valdymo metodas yra labai panašus į rolėmis paremtą prieigos valdymo metodą, Kiekvienas naudotojas turi dinامينius arba statinius atributus, kurių pagalba gaunama prieiga prie skirtos posistemės. Taigi atributai gali būti statiniai arba dinaminiai, tai reiškia arba atributai naudotojui yra nustatyti, arba jie būna priskiriami dinamiška. Priskirti

atributai gali būti kito naudotojo arba administratoriaus. Taip pat gali būti tokie atributai, kuriuos įveda pats naudotojas, kaip kodas ar miestas, ar bet koks kitas elementas pagal, kurį būtų galima atskirti sistemų prieigą.

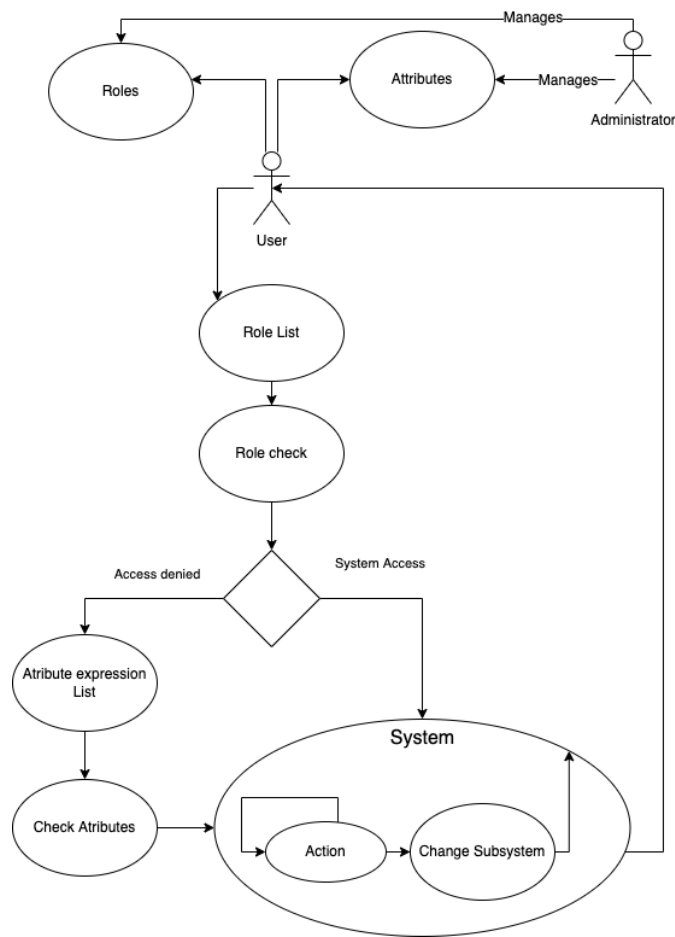


*Pav. 12 Atributais paremta valdymo metodo modelis*

Šitas valdymo metodas yra labai panašus į rolėmis grįstą prieigos valdymo metodą, kadangi tikrinimo principai panašūs. Naudotojų sąrašas privalo būti, statiniai atributai turi būti susietos su posistemėmis per sudarytas atributų taisykles. Yra sudaromos taisyklės kiekvienai posistemei. Taisyklės yra skirtingos pagal atributus ir posistemas. Yra sudaryti posistemių sąrašai su susietomis taisyklėmis. Sąrašai, bet kuriuo metu gali būti atnaujinti. Naudojant priskirtus atributus naudotojams yra patikrinama ar galima gauti prieigą prie norimos posistemės pagal sudarytas taisykles. Dinaminiai atributai yra tikrinami kaip ir statiniai atributai pagal priskirtas posistemių taisykles. Visi duomenys yra saugomi MySQL duombazėje, naudotojų sąrašai, posistemių sąrašai, taisyklių sąrašai, kurie yra susieti su posistemių sąrašais.

### **Atributais-Rolėmis grįstas prieigos valdymo metodas**

Šitas metodas yra sujungtas iš dviejų metodų, kurie buvo minėti anksčiau: rolėmis grįstas prieigos valdymo metodas ir atributais grįstas prieigos valdymo metodas. Atributais ir rolėmis grįstas prieigos metodas yra vienas iš dažnesnių pasirinkimų didesnėms sistemoms. Pastoviams naudotojams, kurie naudoja sistemą ir nereikia papildomos prieigos prie sistemos. Atributų naudojimas yra skirtas prieigai, kuri skirta naudotojams, kuriems reikia prieigos ne vien prie skirtos posistemės. Kadangi šitas metodas naudoja tiek atributus tiek roles sunkesnis padaryti, bet jis yra dinamiškas, todėl jį naudoja didelės sistemos. Šiam metodui reikia Rolių sąrašo, atributų taisyklių sąrašo, naudotojų sąrašo ir posistemių sąrašo, kuris yra susietas su rolių sąrašu ir taisyklių sąrašu. Pirmiausia patikrinimui yra naudojamas rolių tikrinimas, neturini naudotojui rolių yra patikrinami atributai ir atributų taisyklės.



Pav. 13 Atributais-rolėmis paremta metodo modelis

Trumpam sistemos paaiškinimą matome nuotraukoje. Pirmiausia Naudotojas turi roles ir atributus su kuriais yra susietas arba dinamiškai arba statiškai. Naudotojui norint prisijungti prie sistemos turi praeiti verifikaciją. Pirmiausia yra gaunamas rolių sąrašas, kuriame yra išsaugotos rolės. Gautos rolės yra susietos su posistemėmis. Tai reiškia, kad galima patikrinti ar naudotojas turi prieigą prie specifinių posistemų. Naudotojui neturint reikalingų rolių yra tikrinami turimi atributai. Tikrinant atributus yra gaunamas taisyklių sąrašas, kuris yra susietas su posistemėmis, kaip ir rolių sąrašas. Yra tikrinami tiek statiniai naudotojo atributai tiek dinaminiai. Patikrinus atributus naudotojas yra įleidžiamas į posistemę arba neįleidžiamas į posistemę. Valdant prieigą tokiu būdu yra išvengiamos tikrinimo kolizijos, kadangi pirmiausia yra patikrinamos rolės, o vėliau atributai. Žinoma darant tokių būdu atributai ir rolės nėra lygiaverti, bet tai leidžia palengvinti prieigos valdymo metodo sukūrimą ir valdymą, kadangi nereikia išspręsti kolizijų problemos.

Naudotojai	Darbuotojų valdymas	Darbai	Analizė
Jonas	V	V	
Kęstutis		V	V
Petras		V	

Pav. 14 Rolių lentelės pavyzdys

Atributai	Posistemė nr. 1	Posistemė nr. 1
Darbininkas		V
Vairuotojo pažymėjimas		V
Finansai	V	

*Pav. 15 Atributų lentelės pavyzdys*

Pavyzdinė matrica kaip gali atrodyti rolės susietos su naudotojais. Atributai tokios, kaip matricos neturi, kadangi jie yra tikrinami pagal pateiktas taisykles. Taisyklių pavyzdžiai galėtų būti jeigu naudotojas yra darbininkas gali prieiti prie sistemos, arba gali būti jei darbuotojas yra vyras jis gali naudotis vyrų skirtais daiktais.

Naudojant atributais-rolėmis grįstą prieigos problemą yra išsprendžiama dinamiškumo problema kuri kyla naudojant rolėmis grįstą prieigos metodą. Žinoma yra ir kitų problemų, kurios kyla naudojant šitą metodą, kaip rolių ir atributų kolizija. Rolės gali kirstis su atributais todėl reikia sugalvoti, lygiavertį sprendimą arba rolės ar atributai turi būti viršesni vienas už kitą. Mano koncepte buvo pasirinkta, kad rolės yra anksčiau tikrinamos negu atributai, taip yra išvengiama rolių ir atributų kolizija.

## Technologijos

## Išvados

- Išanalizavus žiniatinklio programų prieigos valdymo problemas pastebėta, kad kuriant prieigos valdymo modelį yra ne viena problema su kuria susiduriama. Visos problemos gali būti suskirstytos į tris pagrindines problemas. Pirmiausia tai yra žinomiausia problema duomenų apsaugos problema. Kita viena iš pagrindinių problemų yra naudotojo konfidencialumas, apie kurį reikia pagalvoti. Paskutinė didelė problema yra sistemos prieiga, reikia pagalvoti apie sistemos apkrovą naudojant prieigos valdymo metodą. Žinoma yra ir kitų problemų su kuriomis galima susidurti.
- Išanalizavus žiniatinklio programų prieigos valdymo metodus buvo pastebėta, kad jų yra daug ir skiriasi drastiškai nuo analizuojamo metodo, arba yra metodai yra paremti rolėmis grįsto valdymo metodu.
- Atlikus rolėmis grįsto prieigos valdymo metodo analizę buvo suprasta, kad šitas metodas nėra toks lengvas kaip iš pradžių buvo galvota. Buvo pamatyta, kad jį panaudoti reikia daug žingsnių kuriuos reikia atlikti. Pastebėta kaip veikia naudotojų valdymas, rolių valdymas ir licencijų tvarkymas. Išsiaiškinta, kaip turėtų veikti prieigos valdymas iš administratoriaus pusės.
- Buvo išanalizuota kur žiniatinklyje yra pritaikytas rolėmis grįstas metodas. Buvo pastebėta, kad yra metodų kurie remiasi rolių prieigos valdymo metodus svetainėse, kurios naudoja programinę įrangą susijusią tarp skirtingų puslapių. Taip pat buvo išsiaiškinta kad rolėmis grįstas prieigos valdymo metodas yra naudojamas debesų paslaugoms valdyti.

Buvo išanalizuotas rolėmis grįstas prieigos valdymo metodas, bei jo pritaikymas internete. Taip pat buvo atkreiptas dėmesys ir į kitus metodus, kurie yra taip pat skirti prieigos valdymui.



## Literatūros sąrašas

- [1] D. R. K. R. C. David F.Fraiolo, *Role-Based Access Control*. 2003.
- [2] N. Mundbrod and M. Reichert, “Object-Specific Role-Based Access Control,” *Int. J. Coop. Inf. Syst.*, vol. 28, no. 1, pp. 1–30, 2019, doi: 10.1142/S0218843019500035.
- [3] S. Bhatt, T. K. Pham, M. Gupta, J. Benson, J. Park, and R. Sandhu, “Attribute-Based Access Control for AWS Internet of Things and Secure Industries of the Future,” *IEEE Access*, vol. 9, pp. 107200–107223, 2021, doi: 10.1109/ACCESS.2021.3101218.
- [4] M. Belchior, D. Schwabe, and F. Silva Parreiras, “Role-based access control for model-driven web applications,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 7387 LNCS, pp. 106–120, 2012, doi: 10.1007/978-3-642-31753-8\_8.
- [5] M. U. Aftab *et al.*, “Permission-based separation of duty in dynamic role-based access control model,” *Symmetry (Basel)*, vol. 11, no. 5, 2019, doi: 10.3390/sym11050669.
- [6] J. P. Cruz, Y. Kaji, and N. Yanai, “RBAC-SC: Role-based access control using smart contract,” *IEEE Access*, vol. 6, no. c, pp. 12240–12251, 2018, doi: 10.1109/ACCESS.2018.2812844.
- [7] A. Karp, H. Haury, and M. Davis, “From ABAC to ZBAC: The evolution of access control models,” *5th Eur. Conf. Inf. Manag. Eval. ECIME 2011*, vol. 9, no. 2, pp. 202–211, 2011.
- [8] L. Dongdong, X. Shiliang, Z. Yan, T. Fuxiao, N. Lei, and Z. Jia, “Role-based access control in educational administration system,” *MATEC Web Conf.*, vol. 139, pp. 1–8, 2017, doi: 10.1051/matecconf/201713900120.
- [9] J. Li, Y. Tang, C. Mao, H. Lai, and J. Zhu, “Role based access control for social network sites,” *2009 Jt. Conf. Pervasive Comput. JCPC 2009*, pp. 389–393, 2009, doi: 10.1109/JCPC.2009.5420153.
- [10] W. T. Tsai and Q. Shao, “Role-based access-control using reference ontology in clouds,” *Proc. - 2011 10th Int. Symp. Auton. Decentralized Syst. ISADS 2011*, vol. 2, pp. 121–128, 2011, doi: 10.1109/ISADS.2011.21.