Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 007:004.056
DOI: https://doi.org/10.24867/15BE24Grujcic

EDITOR ZA XACML RBAC PROFIL

XACML RBAC PROFILE EDITOR

Nikola Grujčić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu je analiziran XACML (OASIS eXtensible Access Control Markup Language) sa akcentom na RBAC (Role Based Access Control) profil ovog jezika. Objašnjeni su teorijski koncepti vezani za kontrolu pristupa. Razmatrano je definisanje prava pristupa kroz politike ovog profila, kojim bi se ispunili zahtevi za osnovni i hijerarhijski RBAC. Dat je opis implementirane aplikacije i prikaz reprezentativnih delova koda.*

Ključne reči: *XACML, RBAC, politika, pravilo, uloga, permisija, kontrola pristupa*

Abstract – *This paper analyzes XACML (OASIS eXtensible Access Control Markup Language) with an emphasis on the RBAC (Role Based Access Control) profile of this language. Terms and theoretical concepts related to access control are explained. The definition of access rights through policies was discussed, which would meet the requirements for basic and hierarchical RBAC. A description of the implemented application and a representation of representative parts of the code are given..*

Keywords: *XACML, RBAC, policy, rule, role, permission, access control*

1. UVOD

Kontrola i upravljanje pristupom informacijama je jedan od veoma bitnih problema svih informacionih sistema koji podržavaju veliki broj korisnika. Kako bi podaci ostali validni i kako bi se sprečila njihova zloupotreba, nameću se potrebe za zaštitom i obezbeđivanjem sigurnosti sistema. Jedno od mogućih načina da se ovaj problem reši jeste kontrola pristupa [1]. Ona predstavlja jedan od osnovnih koncepata bezbednosti čija je svrha ograničavanje operacija koje korisnici sistema mogu izvršiti nad podacima. Daljim razvojem kontrole pristupa, nastajali su različiti modeli koji bi definisali na koji način bi određeni korisnik mogao da pristupi određenom resursu sistema i na koji način bi mogao da ga koristi [1]. Jedan od modela kontrole pristupa koji se zasniva na konceptu uloga i privilegija naziva se *Role Based Access Control* (RBAC) [2]. Tema ovog rada je definisanje prava pristupa koristeći XACML *(extensible Access Control Markup Language)* jezik [3], oslanjajući se na RBAC profil ovog jezika. Ovaj rad opisuje veb editor namenjen za definisanje prava pristupa, adresira potencijalne probleme u pisanju prava pristupa na ovaj način i predlaže rešenja za unapređenje implementiranog softvera. u pisanju prava pristupa na ovaj način i predlaže rešenja za unapređenje implementiranog softvera.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Goran Sladić, red. prof.

2. KONTROLA PRISTUPA

Kontrola pristupa je jedan od osnovnih aspekata bezbednosti informacionih sistema [2]. Motivisana je potrebom da se pristup informacijama i dostupnim računarskim resursima i uslugama dozvoli samo ovlašćenim subjektima [4]. U poslednje tri decenije predloženi su različiti modeli kontrole pristupa [4]. Biće izdvojen poseban model kontrole pristupa koji se zasniva na ulogama, pošto je fokus rada usmeren ka ovom modelu. Nakon izlaganja ovih fundamentalnih stavki, analiziraće se krucijalni koncepti i ideje XACML-a.

2.1. Role Base Access Control

Kontrola pristupa zasnovana na ulogama (RBAC) je jedan od najpopularnijih modela kontrole pristupa za poslovne sisteme zbog svoje fleksibilnosti i skalabilnosti. Ovaj model se zasniva na individualnim ulogama određenog korisnika. Uloge su definisane u odnosu na radne odgovornosti osoba i bezbednosne politike organizacije. Kontrola pristupa zasnovana na grupama i ulogama može se uspešno primeniti u organizacijama sa precizno definisanom hijerarhijom vlasti i podelom dužnosti [2]. Centralna ideja RBAC modela jeste da su dozvole (eng. *Permissions*) povezane sa ulogama (eng. *Roles*), a korisnicima su pridružene odgovarajuće uloge. Stvaraju se uloge za različite funkcije posla u organizaciji i korisnicima se dodeljuju uloge na osnovu njihovih odgovornosti i kvalifikacija. Referentni RBAC model je definisan u okviru četiri modela [2, 4]:

1. osnovni RBAC,
2. hijerarhijski RBAC,
3. statičko razdvajanje dužnosti i
4. dinamičko razdvajanje dužnosti.

Osnovni (eng. *Core*) RBAC definiše minimalnu kolekciju RBAC elemenata, skupova elemenata i odnosa između elemenata kako bi se u potpunosti postigao sistem kontrole pristupa zasnovan na ulogama. To uključuje definisanje odnosa između korisnika i uloge, kao i odnosa između dozvola i uloga. Pored toga, uvodi koncept aktivacije uloga kao deo sesije korisnika u računarskom sistemu. Osnovni RBAC je neophodan u bilo kom RBAC sistemu, dok su ostali modeli nezavisni jedan od drugog. Hijerarhijska komponenta RBAC modela dodaje relacije za podržavanje hijerarhije uloga. Treća i četvrta komponenta modela dodaju relacije za postizanje ekskluzivnosti između uloga u skladu sa zadacima korisnika [2, 4].

2.2. Extensible Access Control Markup Language

XACML (eXtensible Access Control Markup Language) je jezički standard za izražavanje kontrole pristupa koji se

zasniva na XML (Extensible Markup Language) jeziku. Osmišljen je sa namerom da izrazi [5]:

- bezbednosne politike,
- zahteve za pristup koji su potrebni za postavljanje upita nad sistemom politika i
- odgovore sistema sa donošenom odlukom o ovlašćenju.

„Proširivi jezik za označavanje kontrole pristupa“ je jedan od načina izražavanja i mehanizama za sprovođenje kontrole pristupa koja se zasniva na atributima (eng. *Attribute-based access control* - ABAC). Razvijen je sa ciljem da se od operativnih okruženja aplikacije izdvoji proces kreiranja politika i proces donošenja odluka o ovlašćenju [6].

Većina operativnih okruženja implementira kontrolu pristupa na različite načine, svaki sa različitim opsegom kontrole i svaki u odnosu na različite tipove operacija i tipove izvora podataka. Ova heterogenost uvodi brojne administrativne izazove, pored samog izazova sprovođenja politike. Administratori su primorani da se izbore sa mnoštvom bezbednosnih domena pri upravljanju politikama pristupa i atributima. Budući da operativna okruženja implementiraju kontrolu pristupa na različite načine, teško je razmenjivati i deliti informacije o kontroli pristupa između različitih okruženja [6]. XACML nastoji da ublaži ove izazove stvaranjem zajedničkog i centralizovanog načina izražavanja svih podataka kontrole pristupa. Na osnovu zahteva za pristup koji dobija od aplikacija, XACML će donositi i sprovoditi odluke [5].

3. XACML RBAC PROFIL

XACML-ov profil za RBAC ispunjava uslove za osnovnu i hijerarhijsku kontrolu pristupa zasnovanu na ulogama [3]. U RBAC profilu XACML-a, telo za omogućavanje uloga (*Role Enablement Authority* - REA) je entitet koji korisnicima dodeljuje atribute i vrednosti, ili omogućava da u toku sesije korisnik poseduje odgovarajuće atribute i vrednosti kojima dokazuje da poseduje određenu ulogu. REA određuje koji skup uloga može biti dodeljene korisniku [3]. U nastavku će biti predstavljene vrste i svrha politika, koje mogu da se kreiraju. Biće objašnjen način kako da se postigne kontrola pristupa na osnovu ovog profila i adresiraće se ograničenja predloženog profila.

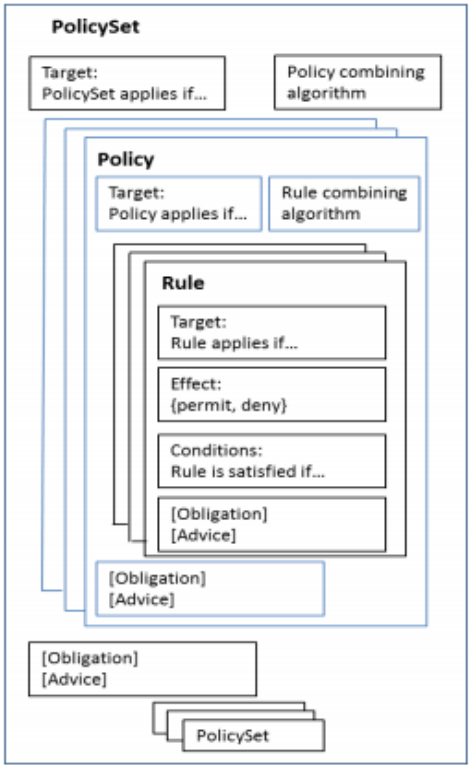
3.1. Politike

Zahtev za pristup se sastoji iz atributa subjekta (uglavnom korisnika koji je izdao zahtev), atributa resursa (resurs za koji se traži pristup), atributa vezanih za akciju (operacije koje treba izvesti na resursu) i atributa vezanih za trenutno okruženje [6]. XACML atributi se specificiraju svojim nazivom i vrednošću, pri čemu vrednosti atributa mogu biti različitih tipova. Svaki atribut označava svojstvo koje se odnosi na subjekat, odnosno resurs, akciju ili na okruženje [6]. Ukoliko uzmemo primer u bankarstvu, atributi uloge se mogu odnositi na vrstu pozicije (npr. blagajnik, službenik, šalterski radnik). Atributi resursa mogu se odnositi na podatke o računima, kreditima itd. Pod okruženjem se može navesti vreme prijema stranke, broj šaltera itd.

Atributi subjekta i resursa se skladište u repozitorijumima. Za razliku od atributa subjekata i resursa koji se moraju uneti administrativnim putem, atributi okruženja zavise od dostupnosti sistemskih senzora i mogu se odnositi na trenutno vreme, dan u nedelji, a može predstavljati i nivo pretnje na sistem.

Na slici 1 ilustrovan je sadržaj XACML politike. Politike pristupa su strukturirane kao skupovi politika. U ovom skupu se mogu naći pojedinačne politike ili drugi skupovi politika. Pravilo je elementarna jedinica politike i ne može da egzistira nezavisno od politike [5]. Budući da nisu sva pravila, politike ili skupovi politika relevantni za svaki zahtev, XACML uključuje element *Target*. Ovaj element definiše jednostavan logički uslov kojim se utvrđuje primenljivost datog skupa politika, odnosno pojedinačne politike ili pravila na konkretni zahtev korisnika [5]. Radi jednostavnosti i razumljivosti, u radu će se pod pojmom politika, pored pojedinačne politike, podrazumevati i skup politika.

Pored *Target* elementa, pravilo uključuje i niz logičkih uslova na osnovu kojih se određuje efekat pravila, tj. odobrava ili odbija zahtev. Svi logički uslovi moraju biti zadovoljeni kako bi se zahtev odobrio. Logički uslovi pravila mogu da budu kompleksniji od uslova u *Target* elementima, jer mogu uključivati složenije funkcije, poput funkcija poređenja vrednosti i kombinacija drugih funkcija. Navode se u *Condition* elementu [5].



Slika 1. *XACML politika* [6]

3.2. Vrste politika

Postoje četiri vrste politika koje su specificirane u ovom profilu. To su [3, 7]:

- *Role <PolicySet>* - politika koja povezuje vlasnike datog atributa i vrednosti uloge sa permisijama za tu ulogu. Svaka ova politika upućuje na jednu odgovarajuću politiku sa

skupom permisija, ali ne sadrži niti upućuje na druge tipove politika.

- *Permission* <*PolicySet*> - politika koja sadrži dozvole povezane sa datom ulogom. Sadrži druge politike kojima se opisuju resursi kojima subjekti imaju dozvoljen pristup. Takođe, precizira i sve uslove tog pristupa i akcije koje se dozvoljene. Data politika može da sadrži reference na druge politike ovog tipa koje su povezane sa drugim ulogama. Na ovaj način se dozvoljava da politika nasledi sve dozvole povezane sa drugom ulogom.
- *HasPrivilegesOfRole* <*Policy*> - politika koja može biti sadržana u *Permission* politici. Ova politika podržava zahteve koji pitaju da li subjekat ima privilegije date uloge.
- *Role Assignment* <*Policy*> ili <*PolicySet*> - politika koja definiše koje se uloge mogu omogućiti ili dodeliti određenom subjektu. Ovom politikom se mogu specificirati ograničenja na različite kombinacije uloga ili ukupan broj uloga koje subjekat može da poseduje.

Politike navedene u ovom profilu mogu odgovoriti na dve vrste pitanja [3]:

- Ako subjekat ima omogućene uloge R1, R2, ... Rn, može li subjekat izvršiti datu akciju nad datim resursom?
- Ako subjekat ima omogućene uloge R1, R2, ... Rn, znači li to da će subjekat imati dozvole povezane sa datom ulogom R'? Odnosno, da li je uloga R' jednaka ili junior od bilo koje od uloga R1, R2, ... Rn?

Dolazimo do zaključka da politike ovog profila ne odgovaraju na pitanje: „Koji skup uloga ima subjekat?“ To pitanje mora rešavati telo za omogućavanje uloga. Pretpostavlja se da su subjektu sve uloge već omogućene u vreme kada se traži odluka o autorizaciji. Takođe, ne bave se okruženjem u kojem se uloge moraju dinamički omogućiti na osnovu resursa ili radnji koje subjekat pokušava da izvrši. Zaključujemo da politike navedene u ovom profilu se ne bave statičkim ili dinamičkim razdvajanjem dužnosti, koji je predviđen RBAC modelom [3].

3.3. Kontrola pristupa

Kontrola pristupa zasnovana na ulogama može da se implementira oslanjajući se na dva tipa politika [3, 7]:

- *Role* politike i
- *Permission* politike.

Potrebno je da za svaku ulogu bude definisana jedna *Role* politika koji sadrži *Target* element koji čini ovu politiku primenljivom samo na subjekte koji imaju XACML atribut pridružen datoj ulozi. Element *Target* ne sme ni na koji način da ograniči resurs, akciju nad resursom ili okruženje [3]. Svaka *Role* politika treba da sadrži jedan *PolicySetIdReference* element koji upućuje na *Permission* politiku sa dozvolama date uloge. Pored toga, *Role* politika ne sme da sadrži nijednu drugu politiku ili da referencira druge politike [3]. Za svaku ulogu treba da se definiše jedna *Permission* politika. Ova politika treba da sadrži *Rule* elemente kojim se specificiraju tipovi pristupa dozvoljeni subjektima koji imaju datu ulogu [3, 7].

4. SPECIFIKACIJA SISTEMA

Softversko rešenje predstavlja veb editor u kojem se definišu prava pristupa oslanjajući se na RBAC profil XACML-a. Osnovni zadatak ovog sistema je da obezbedi proces kreiranja politika kojima bi se zadovoljili zahtevi za definisanje osnovnog i hijerarhijskog RBAC-a. Rešenje se bazira na specifikaciji standarda XACML verzije 3.0 (XACML v3.0 Core and Hierarchical Role Based Access Control (RBAC) Profile Version 1.0) [3].

4.1. Arhitektura sistema

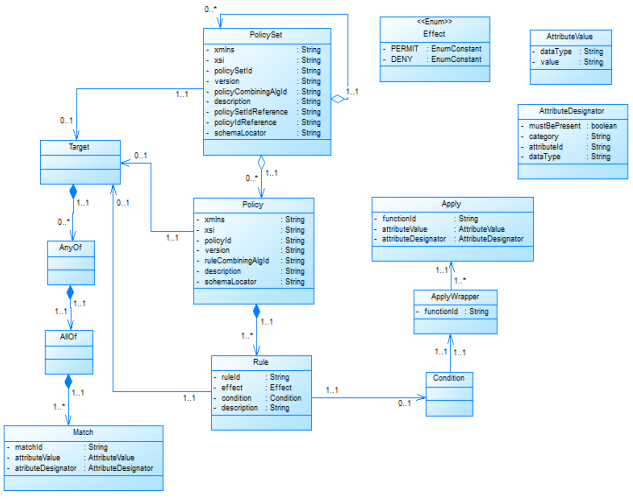
Sistem za kreiranje i editovanje XACML politika se sastoji iz dva modula:

1. korisnički modul i
2. *back-end* aplikaciju.

Korisnički modul predstavlja *front-end* aplikaciju koja obezbeđuje interfejs za krajnjeg korisnika i omogućuje mu sve neophodne funkcionalnosti. Predstavlja *single-page* aplikaciju koja je realizovana upotrebom *Angular* okvira 8.0.6. Poslovna logika sistema je smeštena u *back-end* modulu. Realizovan je kao servisno orijentisana veb aplikacija koja je implementirana u *Java Spring* okviru. Komunikacija između klijentskog i serverskog modula se obavlja putem REST tehnologija. Sav saobraćaj vršen je posredstvom HTTP protokola. *Hibernate* biblioteka se koristi za manipulaciju podacima koji su smešteni u *MySQL* bazi. Pored relacione baze podataka, korišćena je i *MongoDB* baza podataka, kako bi bili ispunjeni zahtevi čuvanja politika u vidu dokumenata. Na *back-end* aplikaciji se koristi *Spring Security* okvir kojim se obavlja autentifikacija korisnika.

4.2. Model podataka

Kako bi se stekao bolji uvid u način organizacije i funkcionisanja sistema, na slici 2 je kroz dijagram klasa prikazan odnos između relevantnih entiteta koji su nam potrebni kako bismo kreirali željene politike.



Slika 2. *Isečak iz dijagrama klasa*

Tek nakon parsiranja u XML format, politike će biti sačuvane u nerelacionoj bazi podataka. *PolicySet* entitet će se sastojati iz instanci *Policy* entiteta, a pored toga može obuhvatati i druge instance *PolicySet* entiteta. *Policy* se sastoji iz *Rule* entiteta. Instance *Policy* entiteta će moći da nastave da egzistiraju ukoliko *PolicySet* više ne bude postojao.

Nasuprot tome, instance *Rule* klase neće moći da egzistiraju ukoliko se obriše roditeljska instanca *Policy* entiteta. Svaki od prethodno navedenih entiteta će sadržati *Target* element. Ovaj element može biti prazan ili može sadržati *AnyOf* instance. Dalje, *AnyOf* instanca mora da sadrži *AllOf* instancu, koja je neophodno da sadrži *Match* instancu. *Rule* entitet sadrži *Condition*. Ovaj element će biti sačinjen iz *Apply* elemenata.

5. IMPLEMENTACIJA SISTEMA

U ovom poglavlju će biti opisane funkcionalnosti čija implementacija je bila neophodna, kako bi se omogućio proces kreiranja i editovanja željene politike.

Nakon uspešne registracije i prijave na sistem, korisnik će imati mogućnost kreiranja politika. Ova proces će se sastojati iz nekoliko *Reactive* formi, kako bi se uneli svi potrebni podaci i kako bi interakcija korisnika sa sistemom bila prirodna i jasna.

Kako bi podaci, prilikom kreiranja politika, bili dostupni unutar aplikacije sa centralizovanog mesta, na klijentskoj strani je potrebno koristiti odgovarajući sistem za upravljanje stanjem aplikacije. Ovaj problem je rešen pomoću *Store* biblioteke, koju obezbeđuje *NgRx* (*Angular Reactive Extensions*). Na ovaj način je omogućena obrada i skladištenje podataka na centralizovanom mestu unutar aplikacije i praćenje promena prouzrokovanih događajima. Radi lakšeg snalaženja i bolje preglednosti podataka, a i pošto je XML moguće vizualizovati kroz strukturu stabla, autor se opredelio da koristi *TreeGrid* komponentu iz *Syncfusion Angular UI* biblioteke komponenata. Ovom komponentom je omogućeno dinamičko dodavanje elemenata u stablo, proširivanje i skupljanje zapisa prikazanih u stablu, što odgovara potrebama projekta.

Kako bismo zapisali politiku u XML formatu, potrebno je da se koristi odgovarajući parser. Izabran je JAXB (*Java Architecture for XML Binding*) parser.

U nastavku će biti prikazana politika koja je kreirana implementiranim rešenjem. Pretpostavimo da bi u organizaciji postojala uloga *employee* koja bi imala dozvolu za čitanje politika. Neophodno je definisati jednu *Role* i *Permission* politiku. *Permission* politika koja sadrži permisiju vezanu za datu ulogu je prikazana na slici 3.

```
<PolicySet PolicyCombiningAlgId="&policy-combine:permit-overrides"
    Version="0.1"
    PolicySetId="PPS:employee:role"
    xsi:schemaLocator="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17
        xacml-core-v3-schema-wd-17.xsd"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17">
    <Policy Version="1.0" xsi:schemaLocator="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17
        xacml-core-v3-schema-wd-17.xsd"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17"
        RuleCombiningAlgId="&rule-combine:permit-overrides"
        PolicyId="Permissions:of:employee:role">
        <Rule Effect="Permit" RuleId="Permission:to:read:policy">
            <Target>
                <AnyOf>
                    <AllOf>
                        <Match MatchId="&function:string-equal">
                            <AttributeValue
                                DataType="&xml:string">read policy</AttributeValue>
                            <AtributeDesignator
                                DataType="&xml:string"
                                AttributeId="&action:action-id"
                                Category="&category:action"
                                MustBePresent="false"/>
                        </Match>
                    </AllOf>
                </AnyOf>
            </Target>
        </Rule>
        <Target/>
    </Policy>
    <Target/>
</PolicySet>
```

Slika 3. *Primer Permission politike*

Ova permisija će dozvoliti bilo kojem subjektu da obavi akciju čitanja politike, ukoliko se dodeli odgovarajućoj *Role* politici.

6. ZAKLJUČAK

U radu je predloženo softversko rešenje za definisanje prava pristupa putem politika, oslanjajući se na RBAC profil XACML jezika. Na osnovu ovih politika i uz pomoć odgovarajućeg mehanizma za sprovođenje kontrole, moguće je obezbediti kontrolu pristupa u informacionim sistemima. Obrađene su osnovne ideje na kojim se bazira kontrola pristupa, analiziran je XACML jezik i njegov mehanizam sprovođenja kontrole pristupa. Ove teorijske koncepte je bilo neophodno izložiti pre samog analiziranja RBAC profila XACML-a. Takođe, opisano je implementirano softversko rešenje i prikazana je njegova arhitektura.

Trenutno implementiranim rešenjem je ograničen pristup i editovanje politike na kreatora politike. Jedno od mogućih proširenja ovog rešenja jeste podrška da više korisnika jedne organizacije mogu da kreiraju i preuređuju politike. Ukoliko bi se ovo proširenje podržalo, bilo bi neophodno uvesti i podršku za verzioniranje politika. Verzioniranjem bi se obezbedilo da stanje politike uvek ostane konzistentno i osigurala bi se potpuna kontrola nad procesom kreiranja politika. Takođe, korisniku bi bila dostupna i istorija svih izmena nad politikama.

7. LITERATURA

[1] A. Poniszewska-Maranda, Management of access control in information system based on role concept, 2011.

[2] D. R. K. R. C. David F. Ferraiolo, Role-Based Access Control - Second edition, 2007.

[3] H. Bill Parducci, XACML v3.0 Core and Hierarchical Role Based Access Control RBAC) Profile Version 1.0 Committee Specification 02, 2014.

[4] S. I. T. I. Council, American National Standard for Information Technology – Role Based Access Control, 2004.

[5] The eXtensible Access Control Markup Language (XACML), Version 3.0, OASIS Standard, 2013.

[6] R. C. R. K. a. V. H. David Ferraiolo, Extensible Access Control Markup Language (XACML) and Next Generation Access Control (NGAC), 2016.

[7] S. M. Anne Anderson, Core and Hierarchical Role Based Access Control (RBAC) profile of XACML v2.0, 2005.

Kratka biografija:

Nikola Grujčić rođen je je u Zrenjaninu 1996. godine. Osnovne akademske studije završio je 2019. godine na Fakultetu tehničkih nauka. Master rad iz oblasti Elektrotehnike i računarstva – Računarstvo i automatika je odbranio 2021. godine

kontakt: nikolagrujcic@gmail.com