Informaciona bezbednost

Kontrola pristupa - Role Based Access Control

Goran Sladić

Katedra za informatiku

2025.

Fakultet tehničkih nauka
Univerzitet u Novom Sadu

Role Based Access Control

- Uloga (*role*) $\sim$ radno mesto u organizaciji
- Razvoj krajem 1980-tih i početkom 1990-tih
  - Dobson-McDermid - zovu ih funkcionalne uloge
  - Baldwin - zove ih *Named Protection Domains* (NDS)
  - Nash-Poland - kako koristiti uloge prilikom autentifikacije uređaja u bankarskoj industriji
- 1992. NIST studija
  - Stanje u privatnom i javnom sektoru – koristi se DAC
- DAC nije najbolje prilagođen potrebama:
  - Stvarni vlasnik podataka nije korisnik već organizacija
  - Diskreciona kontrola nad delom korisnika čak i nije prikladna
- Konvencionalni MAC ne odgovara potrebama
  - Potrebna je kontrola pristupa bazirana na kompetenciji
  - Sprečavanje konflikta interesa

Role Based Access Control

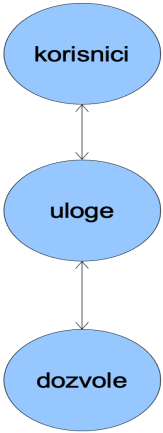
- Inicijalni RBAC model: Ferraiolo-Kuhn 1992.
- Tri pravila
  1. Dodela uloga
     - Subjekat može da izvrši transakciju samo ako mu je dodeljena uloga ili je izabrao neku ulogu
     - Autentifikacija nije transakcija
     - Sve posle autentifikacije se izvršava u obliku transakcija za koje je potrebno da subjekat ima aktivnu ulogu
  2. Autorizacija uloga
     - Subjekat može da koristi samo uloge koje su mu autorizovane
  3. Autorizacija transakcija
     - Subjekat može da izvršava transakciju samo ako je transakcija autorizovana za korisnikovu aktivnu ulogu

Role Based Access Control

- Formalna definicija pravila
  - Aktivna uloga subjekta s – $AR(s : subject)$
  - Skup uloga koje može izabrati s – $RA(s : subject)$
  - Skup transakcija koje može pokrenuti uloga r – $TA(r : role)$
  - Predikat $exec(s : subject, t : transaction) = true$ akko subjekat s može da pozove transakciju t
- Tri pravila
  - Dodela uloga – $\forall s : subject, t : transaction \cdot exec(s, t) \Rightarrow AR(s) \neq \emptyset$
  - Autorizacija uloga – $\forall s : subject \cdot AR(s) \subseteq RA(s)$
  - Autorizacija transakcija – $\forall s : subject, t : transaction \cdot exec(s, t) \Rightarrow t \in TA(AR(s))$
    - Implikacija (ne ekvivalencija) omogućava da se uvedu dodatna ograničenja na mogućnost pozivanja transakcija

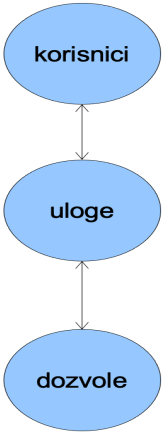
Role Based Access Control

- Uloga predstavlja skup dozvola
- Korisnicima se dodeljuje jedna ili više uloga
- Uloga $\neq$ Grupa
  - Uloga je skup dozvola
  - Grupa je skup korisnika
- Administracija uključuje dva tipa veza
  - Veza između korisnika i uloga
  - Veza između uloga i dozvola



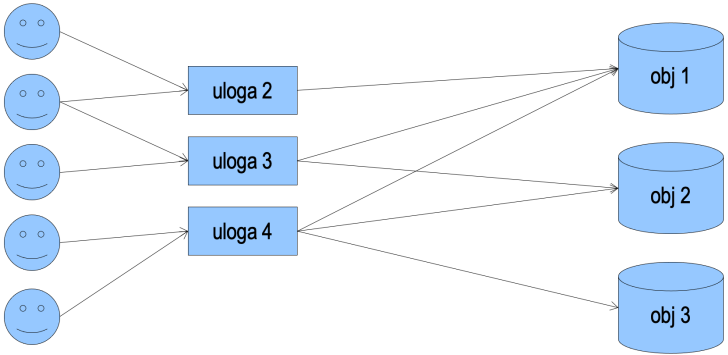
Role Based Access Control

- Kada korisnik promeni poziciju na poslu, samo se veza korisnik-uloga menja
- Ako je pozicija na poslu predstavljena jednom ulogom, kada se se pozicija promeni dve korisnik-uloga veze se menjaju
  - Treba ukloniti vezu između korisnika i trenutne uloge
  - Treba dodati vezu između korisnika i nove uloge



Role Based Access Control

- U okviru organizacije uloge se retko menjaju
- Dok se korisnici i dozvole mogu menjati češće
- Pojednostavljena administracija



Role Based Access Control

- Postoji direktna veza između cene administracije i broja veza koje se moraju održavati za kontrolu pristupa
- Što je veći broj veza, administracija je skuplja i podložnija greškama
- RBAC predstavlja uštedu u administrativnim aktivnostima:
  - Veza između dozvola i korisnika se može opisati kao uređeni par (U, P)
  - U – Skup korisnika na istom radnom mestu
  - P – Skup dozvola potrebnih za obavljanje zadataka na tom radnom mestu
  - Broj veza potrebnih za direktno povezivanje svih korisnika i njihovih dozvola – $|U| \cdot |P|$
    - $|U|$ – broj korisnika u skupu U
    - $|P|$ – broj dozvola u skupu P
  - Tj. za svakog korisnika u U, postoji veza sa svakom dozvolom u P

Role Based Access Control

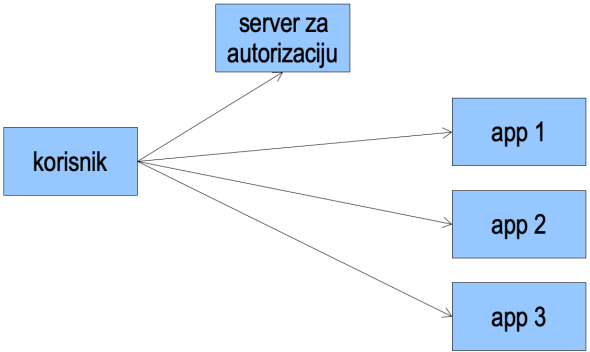
- Uloga se može opisati kao skup dozvola
- Skup P se može interpretirati kao skup uloga ili pozicija na poslu čije su veze korisnik-uloga i uloga-dozvola predstavljene kao uređeni par (U, P)
- Broj veza korisnik-uloga i uloga-dozvole je $|U| + |P|$
- Tj. veza sa svakom ulogom P za svakog korisnika u U i veza sa ulogom P za svaku dozvolu u P
- Za poziciju na poslu, ako je $|U| + |P| < |U| \cdot |P|$
- Tada postoji ušteda u administraciji direktnog povezivanja dozvola za korisnike ako je $|U|, |P| > 2$
- Ako ima ukupno n radnih mesta, ušteda postoji kada je $\sum_{i=1}^{n}(|U_i| + |P_i|) < \sum_{i=1}^{n}(|U_i| \cdot |P_i|)$

Role Based Access Control

- Administracija je obično centralizovana
  - Na jednom serveru su definisane uloge, dozvole i korisnici
- Dva pristupa centralizaciji
  - *User pull*
  - *Server pull*

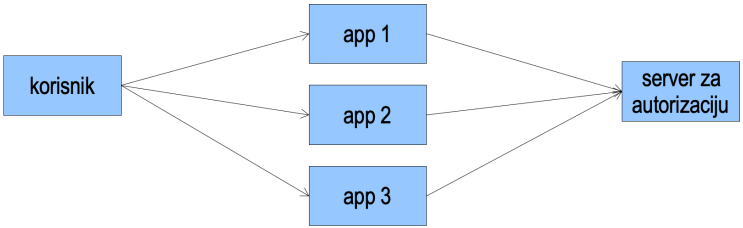
Role Based Access Control

- *User pull* centralizovana autorizacija
  - Korisnik se prijavljuje na server za autorizaciju
  - Od njega dobija neke podatke kojima će se predstaviti aplikacijama
  - Prilikom obraćanja aplikacijama dostavlja i te podatke



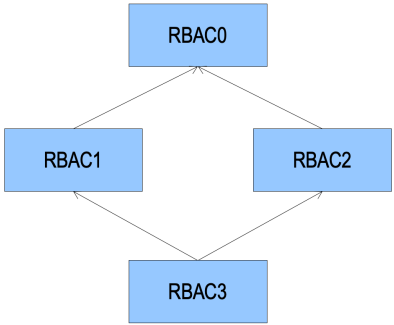
Role Based Access Control

- *Server pull* centralizovana autorizacija
  - Aplikacije su zadužene za autentifikaciju
  - Podaci o pravima su centralizovani na serveru
  - Kada korisnik pristupi aplikaciji, aplikacija se obraća serveru radi dobijanja njegovih dozvola



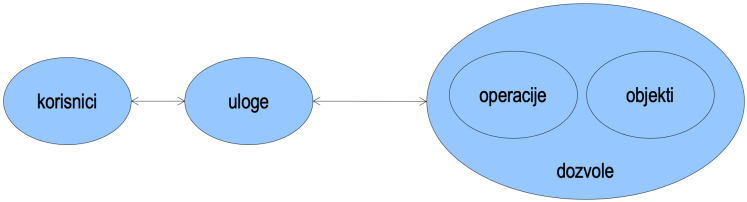
Role Based Access Control

- Sandhu 1996: četiri varijante RBAC-a
  - RBAC0: osnovni elementi RBAC sistema
  - RBAC1: RBAC0 + hijerarhije uloga
  - RBAC2: RBAC0 + ograničenja (*Separation of Duties* (SoD))
  - RBAC3: RBAC0 + RBAC1 + RBAC2



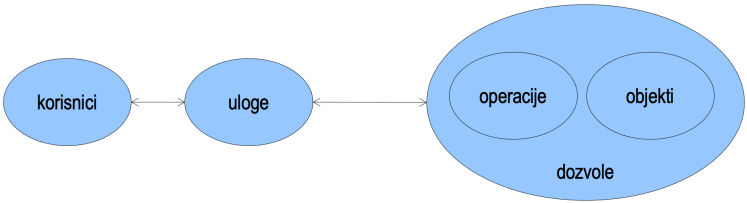
Osnovni RBAC

- Pet koncepata
  - Korisnici
  - Uloge
  - Dozvole, koje se sastoje iz
    - Operacija dopuštenih nad
    - Objektima



Osnovni RBAC

- Veze su n:m
  - Korisnik može imati više uloga
  - Uloga može imati više korisnika
  - Uloga može imati više dozvola
  - Dozvola može biti u više uloga



Osnovni RBAC

- Granularnost dozvola se može birati prema potrebama
  - Dozvola se može posmatrati kao atomična operacija u sistemu
  - Operacije mogu biti implementirane kao transakcije
- Primer: šalterski radnik u banci
  - Može da isplati novac (*withdraw*) ili da uplati novac na račun (*deposit*)
    - Trebaju mu *read* i *write* prava za podatke o računima
  - Ne može da ispravlja ništa nakon obavljene transakcije
- Primer: supervizor u banci
  - Može da ispravi rezultat neke transakcije
    - Trebaju mu *read* i *write* prava za podatke o računima
  - Ali ne može samostalno da pozove *withdraw* ili *deposit*

Osnovni RBAC

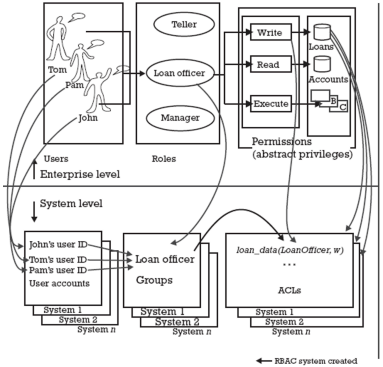
- Veza korisnik - subjekat je 1:n
  - Subjekat je tipično aktivni entitet - npr. program
  - Subjekat se vezuje za jednu sesiju
  - Korisnik može imati više istovremenih subjekata
  - Korisnik može imati više uloga
  - Subjekat može aktivirati podskup mogućih uloga dodeljenih korisniku radi ispunjenja principa minimalnih privilegija
  - Mogućnost izbora aktivnih uloga za subjekta predstavlja dinamičku komponentu osnovnog RBAC-a

Osnovni RBAC

- RBAC predstavlja apstraktan model
- Njegova implementacija zavisi od korišćene tehnologije
- Koncepti RBAC-a moraju se mapirati na koncepte sistema
  - Prava pristupa fajlovima, korisnici, grupe, itd.

Osnovni RBAC

- Mapiranje apstraktnih dozvola iz poslovnog modela na stvarne ACL liste na nivou sistema



Slika preuzeta iz: *Role-Based Access Control*, David F. Ferraiolo, D. Richard Kuhn, Ramaswamy Chandramouli, Artech House, 2003.

Hijerarhijski RBAC

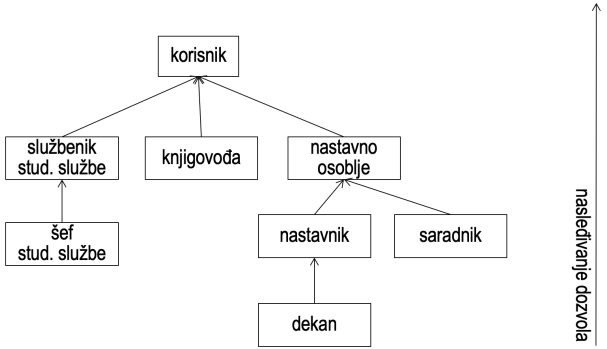
- Hijerarhija uloga
  - Uloge mogu da nasleđuju dozvole od drugih uloga
  - Ako uloga A nasleđuje ulogu B, tada sve dozvole koje ima uloga B pripadaju i ulozi A
    - Tj. skup dozvola u B je podskup skupa dozvola u A

Hijerarhijski RBAC

- Hijerarhije uloga su prirodno sredstvo za strukturiranje uloga tako da odslikavaju organizaciju, raspodelu zaduženja i odgovornosti
- Iako inicijalno zahteva složeniju pripremu, korišćenje hijerarhije uloga se isplati na duži rok kroz jednostavniju administraciju

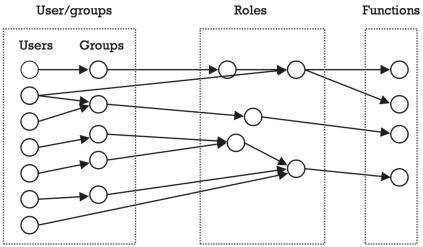
Hijerarhijski RBAC

- Motivacija za formiranje hijerarhije uloga: uloge u organizaciji često imaju preklapajuće funkcije



Hijerarhijski RBAC

- Šeme nasleđivanja - direktno nasleđivanje dozvola
  - Uloga je imenovani skup dozvola
  - Uloga r_2 nasleđuje ulogu r_1 ako je skup dozvola r_1 podskup skupa dozvola r_2
  - Korisnici (i grupe korisnika) definišu se i admninistriraju odvojeno od uloga
  - Baldwinov graf privilegija

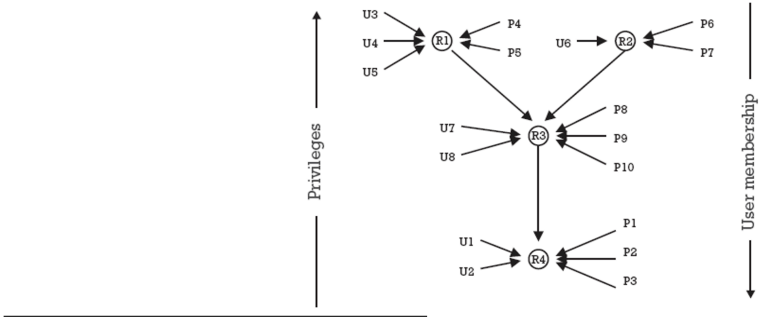


mogućnost redundanse: korisnik može dobiti ulogu direktno ili preko grupe (grupe se administriraju odvojeno od RBAC-a!)

Slika preuzeta iz: *Role-Based Access Control*, David F. Ferraiolo, D. Richard Kuhn, Ramaswamy Chandramouli, Artech House, 2003.

Hijerarhijski RBAC

- Šeme nasleđivanja - nasleđivanje dozvola i korisnika
  - Uloga obuhvata i dozvole i korisnike
  - Uloga ovde služi kao korisnik sa jedne strane i kao kolekcija dozvola sa druge
  - Uloge pri vrhu hijerarhije su ”jače” (imaju više dozvola i manje korisnika)
  - Npr. korisnik U3 ima dozvole P1-P5, P8-P10



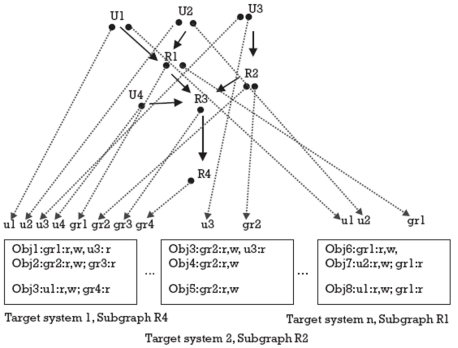
Slika preuzeta iz: *Role-Based Access Control*, David F. Ferraiolo, D. Richard Kuhn, Ramaswamy Chandramouli, Artech House, 2003.

Hijerarhijski RBAC

- Šeme nasleđivanja - zadržavanje korisnika i indirektno nasleđivanje dozvola
  - Dozvole se dodeljuju grupama korisnika
  - Grupe se mapiraju na uloge
  - Uloge su vezane u hijerarhiju
  - Upravljanje hijerarhijama uloga se svodi na upravljanje relacijama nad korisnicima
  - Uloga r_1 „sadrži“ ulogu r_2 ako svi korisnici koji imaju r_1 imaju i r_2
  - Dodeljivanje uloge r korisniku obuhvata
    - Dodeljivanje korisnika svim grupama koje se mapiraju na r
    - I dodeljivanje korisnika svim grupama koje se mapiraju na role koje r sadrži

Hijerarhijski RBAC

- Šeme nasleđivanja - zadržavanje korisnika i indirektno nasleđivanje dozvola
- Primer



Apstraktni korisnici U1-U4 se implementiraju u sistemu kao u1-u4.

Grupe gr1-gr4 odgovaraju ulogama R1-R4.

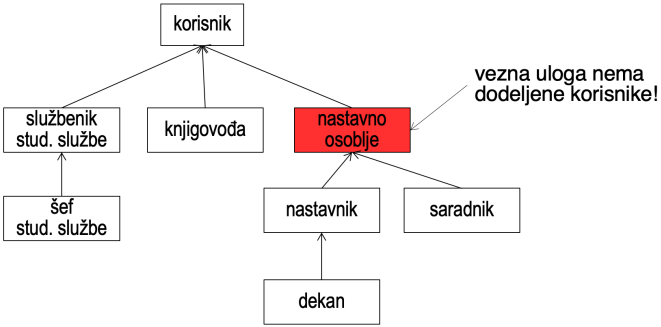
Grupe sadrže sledeće:
gr1 ima u1 i u2
gr2 ima u2
gr3 ima u1-u4
gr4 ima u1-u4

Formalno ne postoji nasleđivanje dozvola, ali se nasleđivanjem korisnika postiže ekvivalentan rezultat!

Slika preuzeta iz: *Role-Based Access Control*, David F. Ferraiolo, D. Richard Kuhn, Ramaswamy Chandramouli, Artech House, 2003.

Hijerarhijski RBAC

- Vezne uloge $\sim$ apstraktne klase
  - Zgodno za grupisanje dozvola
  - Za ograničavanje nasleđenih dozvola

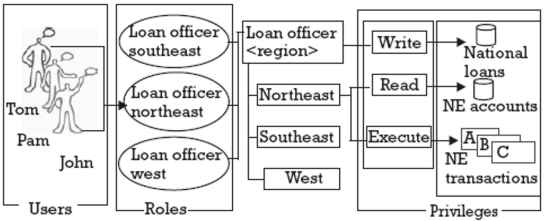


Hijerarhijski RBAC

- Više hijerarhija u jednoj organizaciji
  - Organizaciona hijerarhija
  - Teritorijalna hijerarhija
- Npr. šalterski službenik ima read/write prava nad podacima o bankovnim računima, ali samo za korisnike koji su otvorili račun u jednoj filijali
- Morali bismo definisati posebne uloge za jedno isto radno mesto ali za svaku filijalu posebno! – nepraktično
- Uvodimo **tipove uloga** (*role types*): uloga koja ima kvalifikator
  - Kvalifikatori: lokacija, org. jedinica, region, itd.

Hijerarhijski RBAC

- Tipovi uloga
- Primer
  - Tip uloge: kreditni savetnik
  - Konkretne uloge: kreditni savetnik vezan za konkretni region
  - Globalnim podacima o kreditima mogu svi da pristupaju, a lokalnim samo pojedine uloge



Slika preuzeta iz: *Role-Based Access Control*, David F. Ferraiolo, D. Richard Kuhn, Ramaswamy Chandramouli, Artech House, 2003.

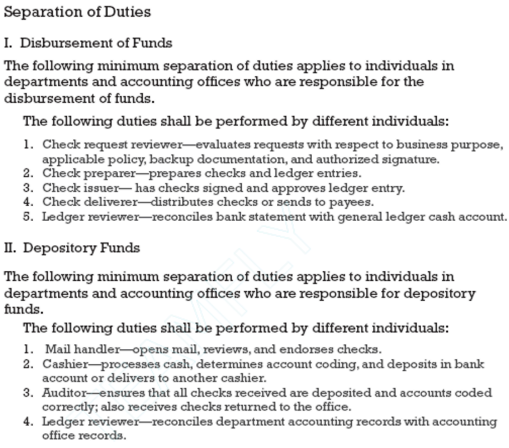
Hijerarhijski RBAC

- Ograničene (*limited*) hijerarhije $\sim$ jednostruko nasleđivanje
- Opšte (*general*) hijerarhije $\sim$ višestruko nasleđivanje
- Ograničene su mnogo raširenije u komercijalnim proizvodima

RBAC sa ograničenjima

- Razdvajanje zaduženja (SoD): kritične operacije obavljaju dva ili više lica, tako da bezbednost ne zavisi od jedne osobe
  - Otvaranje sefa u banci: klijent i službenik
  - „Two-man rule“ za aktivaciju nuklearnog oružja
  - Uplata i isplata u knjigovodstvu

RBAC sa ograničenjima



Separation of Duties

I. Disbursement of Funds

The following minimum separation of duties applies to individuals in departments and accounting offices who are responsible for the disbursement of funds.

The following duties shall be performed by different individuals:

1. Check request reviewer—evaluates requests with respect to business purpose, applicable policy, backup documentation, and authorized signature.
2. Check preparer—prepares checks and ledger entries.
3. Check issuer— has checks signed and approves ledger entry.
4. Check deliverer—distributes checks or sends to payees.
5. Ledger reviewer—reconciles bank statement with general ledger cash account.

II. Depository Funds

The following minimum separation of duties applies to individuals in departments and accounting offices who are responsible for depository funds.

The following duties shall be performed by different individuals:

1. Mail handler—opens mail, reviews, and endorses checks.
2. Cashier—processes cash, determines account coding, and deposits in bank account or delivers to another cashier.
3. Auditor—ensures that all checks received are deposited and accounts coded correctly; also receives checks returned to the office.
4. Ledger reviewer—reconciles department accounting records with accounting office records.

Slika preuzeta iz: *Role-Based Access Control*, David F. Ferraiolo, D. Richard Kuhn, Ramaswamy Chandramouli, Artech House, 2003.

RBAC sa ograničenjima

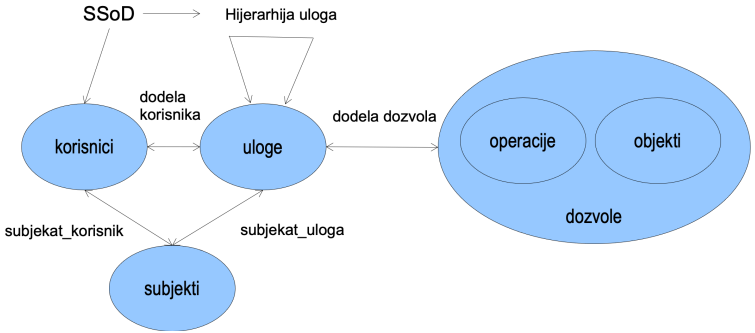
- Statički SoD
  - Ograničenja se postavljaju u trenutku kada se korisniku dodeli uloga
  - Npr. ako je korisniku dodeljena uloga A, ne sme mu biti dodeljena uloga
- Dinamički SoD
  - Ograničenja se postavljaju u trenutku kada korisnik koristi sistem (u toku sesije)
  - Npr. korisnik ne sme imati istovremeno aktivne uloge A i B
  - Obe uloge mu mogu biti statički dodeljene
  - Ali ih ne može imati istovremeno aktivne

RBAC sa ograničenjima

- Definišu se dva zahteva za statički SoD:
  - Statički SoD bez hijerarhije uloga
    - Dodeljivanje jedne uloge može sprečiti dodeljivanje druge, u skladu sa SoD pravilima
  - Statički SoD sa hijerarhijom uloga
    - Dodeljivanje jedne uloge može sprečiti dodeljivanje druge, i svih njenih potomaka

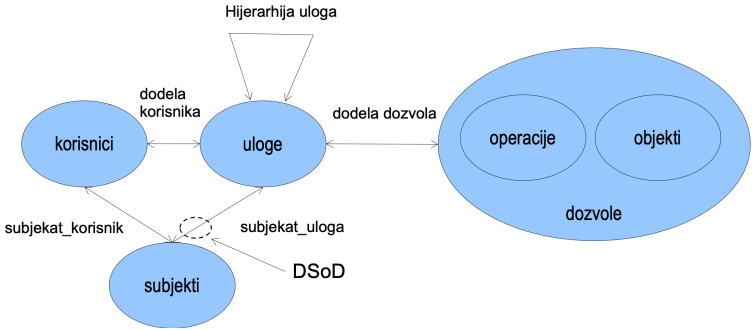
RBAC sa ograničenjima

- Statički SoD
  - Statički SoD: uređeni par (skup uloga, n) gde nijedan korisnik ne može imati više od n uloga iz ovog skupa (obično n=1)



RBAC sa ograničenjima

- Dinamički SoD
  - Ograničenja povezuju subjekte i uloge
- Dinamički SoD: uređeni par (skup uloga, n) gde nijedna korisnička sesija (subjekat) ne može imati više od n uloga iz ovog skupa (obično n=1)



RBAC sa ograničenjima

- SoD baziran na objektima
- Primer: zahtev i odobravanje troškova
  - Statički SoD: nijedna osoba ne sme imati obe uloge
  - Nepraktično za male organizacije
  - Rešenje: nijedna osoba ne sme imati obe uloge **za isti objekat** (isti zahtev za nabavku)
- SoD baziran na istoriji
  - Korisnik može imati sve dozvole za obavljanje kritičnog zadatka (što inače ne bi smeo) ali ne može obaviti **sve delove** zadatka **nad istim objektom**

RBAC sa ograničenjima

- Međusobno isključivanje uloga
- Pomoću skupova uloga
  - Dobijanje jedne uloge iz skupa onemogućava dobijanje drugih uloga iz istog skupa
- Pomoću parova uloga
  - Nepraktično: za n uloga postoji $n(n-1)/2$ mogućih parova

RBAC sa ograničenjima

- Dodeljivanje dozvola ulogama tako da se zadovolji SoD princip – nijedan korisnik ne može samostalno da izvrši kritičan zadatak – ne mora biti jednostavno
- Primer:
  - Tri kritična zadatka: T1, T2, T3
  - Potrebne dozvole: P1, P2, P3
  - Treba dodeliti dozvole ulogama tako da nijedna uloga nema sve dozvole potrebne da izvrši zadatak
  - Dve uloge nisu dovoljne!

	P1	*P2*	*P3*
T1	X	X	—
T2	—	X	X
T3	X	—	X

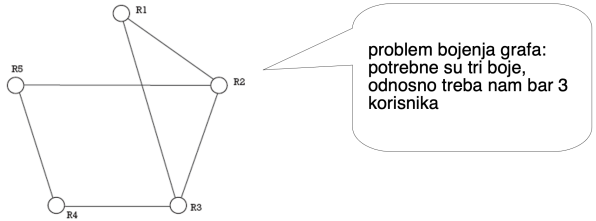
RBAC sa ograničenjima

- Dodeljivanje uloga korisnicima
  - Za dati skup uloga i međusobno isključujućih parova uloga, koliko nam treba različitih korisnika?
- Primer: uloge R1-R5

	R1	*R2*	*R3*	*R4*	*R5*
R1	—	X	X	—	—
R2	X	—	X	—	X
R3	X	X	—	X	—
R4	—	—	X	—	X
R5	—	X		X	—

RBAC sa ograničenjima

- Dodeljivanje uloga korisnicima
  - Za dati skup uloga i međusobno isključujućih parova uloga, koliko nam treba različitih korisnika?
- Primer: uloge R1-R5
- Treba odrediti minimalni broj boja koje su potrebe da se oboje čvorovi grafa tako da nijedna dva susedna temena nisu iste boje
- R1 = crven, R2 = zelen, R3 = plav, R4 = zelen, R5 = crven

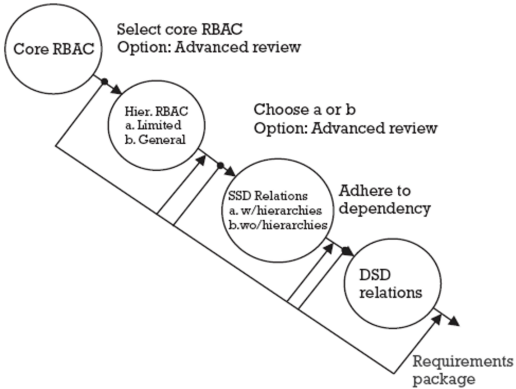


RBAC sa ograničenjima

- Pravila za bezbedno dodeljivanje privilegija u odnosu na SoD
- Potreban uslov: za svaki par međusobno isključivih uloga svaka uloga mora sadržati bar jednu dozvolu koju ne sadrži druga uloga
  - Inače bi jedna uloga bila podskup druge, pa SoD nema smisla
- Dovoljan uslov: za svaki par međusobno isključivih uloga nijedna dozvola iz R1 ne nalazi se u R2
  - SoD je garantovan, ali je ovo često prestrog uslov
- $\Rightarrow$ opšte upozorenje: ako jednu dozvolu sadrži više uloga, postoji potencijalna opasnost za SoD

RBAC standard

- NIST standard
  - RBAC standard



Slika preuzeta iz: *Role-Based Access Control*, David F. Ferraiolo, D. Richard Kuhn, Ramaswamy Chandramouli, Artech House, 2003.

Attribute Based Access Control

- ABAC model definiše ovlašćenja koja predstavljaju uslove nad svojstvima i subjekta i objekata
- ABAC model kontroliše pristup objektima procenjujući pravila u odnosu na atribute entiteta (subjekat i objekat), operacija i okruženja relevantnog za zahtev
- ABAC se oslanja na procenu atributa subjekta, atributa objekta i formalnog odnosa ili pravila kontrole pristupa koji definišu dozvoljene operacije za kombinacije atributa subjekt-objekat u datom okruženju
- ABAC dozvoljava neograničen broj atributa koji se kombinuju da bi se zadovoljilo bilo koje pravilo kontrole pristupa

Attribute Based Access Control

- Tri ključna elementa modela:
  - Atributi – definišu se za entitete u konfiguraciji
  - Model arhitekture – primenjuje se na polise koje vrše kontrolu pristupa
  - Model polisa – definiše ABAC polise

Atributi

- Predstavljaju karakteristike koje definišu specifične aspekte subjekta, objekta, uslova okruženja i/ili zahtevanih operacija koje su unapred definisane i unapred dodeljene od strane relevantnih organa
- Atributi sadrže informacije koje ukazuju na klasu informacija koju daje atribut, ime i vrednost
  - Class = HospitalRecordsAccess
  - Name = PatientInformationAccess
  - Value = BusinessHoursOnly
- Postoje tri tipa atributa:
  - Atributi subjekata
  - Atributi objekata
  - Atributi okruženja

Atributi

- Atributi subjekata
  - Subjekat je aktivan entitet (npr. korisnik, aplikacija, proces ili uređaj) koji uzrokuje protok informacija između objekata ili menja stanje sistema
  - Svaki subjekat ima povezane atribute koji definišu identitet i karakteristike subjekta
  - Takvi atributi mogu uključivati identifikator subjekta, ime, organizaciju, naziv posla i tako dalje
  - Uloga subjekta se takođe može posmatrati kao atribut

Atributi

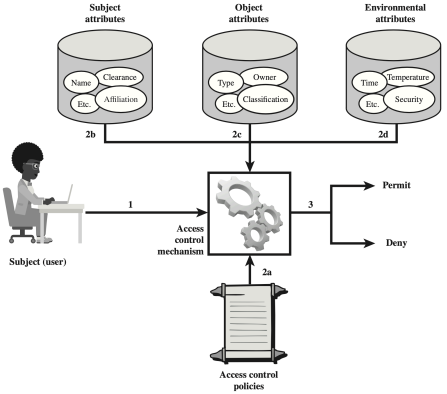
- Atributi objekata
  - Objekat, koji se takođe naziva resurs, je pasivni (u kontekstu datog zahteva) entitet vezan za informacioni sistem (npr. uređaji, datoteke, zapisi, tabele, procesi, programi) koji sadrži ili prima informacije
  - Kao i kod subjekata, objekti imaju atribute koji se mogu iskoristiti za donošenje odluka o kontroli pristupa
  - Npr. Microsoft Word dokument može imati atribute kao što su naslov, predmet, datum i autor
  - Atributi objekta se često mogu izdvojiti iz metapodataka objekta
  - Različiti atributi metapodataka veb servisa mogu biti relevantni za svrhe kontrole pristupa, kao što su vlasništvo, taksonomija servisa ili atributi *Quality of Service* (QoS)

Atributi

- Atributi okruženja
  - Ovi atributi su do sada uglavnom ignorisani u većini smernica kontrole pristupa
  - Oni opisuju operativno, tehničko, pa čak i situaciono okruženje ili kontekst u kojem se dešava pristup informacijama
  - Na primer, atributi, kao što su trenutni datum i vreme ili nivo bezbednosti mreže (npr. Internet naspram intraneta), nisu povezani sa određenim predmetom ili resursom, ali ipak mogu biti relevantni u primenom politike kontrole pristupa

ABAC arhitektura

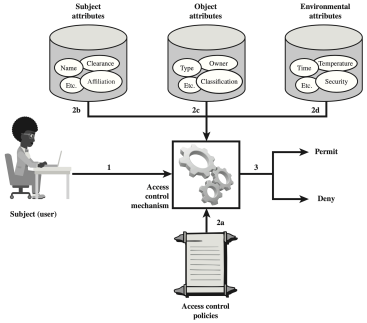
1. Subjekat zahteva pristup objektu. Ovaj zahtev se usmerava ka mehanizmu kontrole pristupa



Slika preuzeta iz: *Computer Security: Principles and Practice*, William Stallings, Lawrie Brown, Pearson, 2018.

ABAC arhitektura

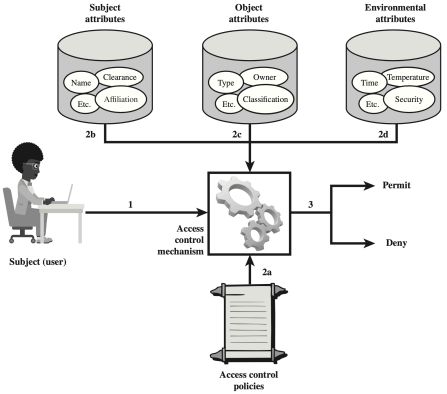
2. Mehanizmom kontrole pristupa upravlja skup pravila (2a) koja su definisana unapred konfigurisanim polisama kontrole pristupa. Na osnovu ovih pravila, mehanizam kontrole pristupa procenjuje atribute subjekta (2b), objekta (2c) i trenutne uslove okruženja (2d) da bi odredio ovlašćenje.



Slika preuzeta iz: *Computer Security: Principles and Practice*, William Stallings, Lawrie Brown, Pearson, 2018.

ABAC arhitektura

3. Mehanizam kontrole pristupa dodeljuje subjektu pristup objektu ako je pristup ovlašćen, i odbija pristup ako nije ovlašćen



Slika preuzeta iz: *Computer Security: Principles and Practice*, William Stallings, Lawrie Brown, Pearson, 2018.

Model polisa

- **Polisa** je skup pravila i odnosa koji regulišu dozvoljeno ponašanje unutar organizacije, na osnovu privilegija subjekata i načina na koji resursi ili objekti treba da budu zaštićeni i pod kojim uslovima okruženja
- **Privilegije** predstavljaju autorizovano ponašanje subjekta
  - Oni su definisani od strane autoriteta i oličeni u polisi
  - Drugi termini koji se obično koriste umesto privilegija su prava i ovlašćenja
- Polisa se obično piše iz perspektive objekta koji treba zaštititi i privilegija dostupnih subjektima

Model polisa

- Model polisa se može definisati na sledeći način:
  1. S, O i E su subjekti, objekti i okruženje
  2. $SA_k(1 \leq k \leq K)$, $OA_m(1 \leq m \leq M)$ i $EA_n(1 \leq n \leq N)$ su predefinisani atributi za subjekte, objekte i okruženje
  3. $ATTR(s)$, $ATTR(o)$ i $ATTR(e)$ su relacije dodele atributa za subjekat s, objekat o i okruženje e:
     - $ATTR(s) \subseteq SA_1 \times SA_2 \times ... \times SA_K$
     - $ATTR(r) \subseteq OA_1 \times OA_2 \times ... \times OA_M$
     - $ATTR(o) \subseteq EA_1 \times EA_2 \times ... \times EA_N$

Model polisa

- Model polisa se može definisati na sledeći način:
  4. U opštem obliku, pravilo polise koje odlučuje o tome da li subjekat s može da pristupi objektu o u određenom okruženju e je Bulova funkcija atributa s, o i e:
     - $Rule : can\_access(s, o, e) \leftarrow f(ATTR(s), ATTR(o), ATTR(e))$
     - Ako je rezultat funkcije tačan, tada se odobrava pristup resursu, u suprotnom pristup se odbija
  5. Baza polisa može se sastojati od brojnih pravila koja pokrivaju mnoge subjekte i objekte unutar domena. Proces odlučivanja o kontroli pristupa svodi se na procenu primenljivih pravila polisa u bazi polisa.

RBAC vs ABAC

- Primer: online streaming platforma koja emituje filmove korisnicima uz fiksnu mesečnu naknadu
- Platforma mora da primeni sledeća pravila:

Movie Rating	Users Allowed Access
R	Age 17 and older
PG-13	Age 13 and older
G	Everyone

RBAC vs ABAC

- U RBAC modelu, svakom korisniku bi tokom registracije bila dodeljena jedna od tri uloge:
  - Odrasla osoba
  - Maloletnik
  - Dete
- Kreirane bi bile tri dozvole
  - Može da gleda filmove sa ocenom R
  - Može da gleda filmove sa ocenom PG-13
  - Može da gleda filmove sa ocenom G
- Uloga *Odrasla osoba* dobija sve tri dozvole
- Uloga *Maloletnik* dobija dozvolu za gledanje filmova sa ocenom PG-13 i sa ocenom G
- Uloga *Dete* dobija samo dozvolu za gledanje filmova sa ocenom G
- I dodeljivanje korisnik-uloga i dodeljivanje dozvola-uloga su ručni administrativni zadaci

RBAC vs ABAC

- ABAC model ne zahteva da se eksplicitno definišu uloge
- Umesto toga, da li korisnik u može da pristupi ili pogleda film m (u okruženju e koje se ovde zanemaruje) biće rešeno procenom pravila sledeće polise:
  - $R1 : can\_access(u, m, e) \leftarrow (Age(u) \geq 17 \wedge Rating(m) \in \{R, PG\text{-}13, G\}) \vee$
    $(Age(u) \geq 13 \wedge Age(u) < 17 \wedge Rating(m) \in \{PG\text{-}13, G\}) \vee$
    $(Age(u) < 13 \wedge Rating(m) \in \{G\})$
  - Gde su uzrast i ocena atribut subjekta i atribut objekta, respektivno
- Prednost ABAC modela je u tome što eliminiše definiciju i upravljanje statičkim ulogama, čime se eliminiše potreba za administrativnim zadacima za dodelu veze korisnik-uloga i dodelu veze dozvola-uloga