HRVATSKA AKADEMSKA I ISTRAŽIVAČKA MREŽA
CROATIAN ACADEMIC AND RESEARCH NETWORK

Modeli kontrole pristupa

CCERT–PUBDOC–2008–02–218

CARNet CERT u suradnji s LS&S

Sigurnosni problemi u računalnim programima i operativnim sustavima područje je na kojem CARNet CERT kontinuirano radi.

Rezultat toga rada ovaj je dokument, koji je nastao suradnjom CARNet CERT–a i LS&S–a, a za koji se nadamo se da će Vam koristiti u poboljšanju sigurnosti Vašeg sustava.

CARNet CERT, www.cert.hr – nacionalno središte za **sigurnost računalnih mreža** i sustava.

LS&S, www.lss.hr – laboratorij za sustave i signale pri Zavodu za elektroničke sustave i obradbu informacija Fakulteta elektrotehnike i računarstva Sveučilišta u Zagrebu.

Sadržaj

1. UVOD 4

2. MODELI KONTROLE PRISTUPA I MODELI SIGURNOSTI 5

2.1. MODEL TEMELJEN NA MOGUĆNOSTIMA 5
2.2. LISTE KONTROLE PRISTUPA (ACL) 6
2.3. DISKRECIJSKI MODEL 6
2.4. MANDATORNI MODEL 8
2.5. MODEL TEMELJEN NA ULOGAMA 10

3. PRIMJERI NA OPERACIJSKOM SUSTAVU LINUX 11

3.1. PRIMJER DISKRECIJSKOG MODELA 11
3.2. PRIMJER MANDATORNOG MODELA 13
3.3. USPOREDBA DAC I MAC MODELA 13

4. ZAKLJUČAK 14

5. REFERENCE 14

1. Uvod

Kontrola pristupa je dio sigurnosne politike koja je prisutna u gotovo svakom operacijskom sustavu. U računalnim sustavima mehanizmi kontrole pristupa ograničavaju korisnike i procese u smislu izvođenja različitih operacija nad objektima, kao što su datoteke, segmenti zajedničke memorije, TCP/UDP priključci, itd. Za svaku takvu akciju mehanizmi kontrole pristupa dodjeljuju svakom korisniku posebne ovlasti za obavljanje određenih akcija na računalu. Uobičajeni su nazivi za računalne entitete subjekt i objekt, gdje subjekt može biti korisnik ili računalni proces, a objekti su računalni resursi kojima se ograničava pristup. Osim toga, mehanizmi kontrole pristupa implementiraju identifikaciju, autentikaciju i autorizaciju subjekata.

Mehanizmi kontrole pristupa prisutni su u nekoliko slojeva u računalnim sustavima, od aplikacijskog sloja preko operacijskog sustava, pa sve do sklopovlja računala. Mehanizmi viših slojeva su više izraženi, ali su i podložniji napadima. Razlozi tome su različiti, a neki od njih su kompleksnost sustava i programerske pogreške. Programski paketi i operacijski sustavi mogu biti vrlo složeni pa postoji velika mogućnost pojave sigurnosnih propusta. Kontrola pristupa na računalnim sustavima ima ulogu ograničavanja štete koju napadači ili neoprezni korisnici mogu prouzročiti.

Modeli kontrole pristupa koje koriste moderni sustavi dijele se u dvije grupe:

- modele temeljene na mogućnostima i
- modele temeljene na listama kontrole pristupa (ACL – *Access Control List*).

Osim ove podjele kontrola pristupa dijeli se i prema metodama implementacije na:

- diskrecijski model (DAC – *Discretionary Access Control*),
- mandatorni model (MAC – *Mandatory Access Control*) i
- model grupa i uloga (RBAC – *Role-based access control*).

Spomenuti modeli kontrole pristupa objašnjeni su u idućim poglavljima. Također opisani su i neki primjeri implementacije modela pristupa kod operacijskih sustava Linux.

2. Modeli kontrole pristupa i modeli sigurnosti

Kontrola pristupa definira mehanizam dozvole ili zabrane pristupa računalnim procesima i različitim računalnim resursima, a osim toga omogućuje nadzor nad sustavom u smislu uvida u aktivnosti prijavljenih korisnika te vođenja evidencije o ostvarenim pristupima svih registriranih korisnika. Osnovni je problem računalne sigurnosti nadzor nad situacijama u kojima neki korisnik ili program smije pristupiti nekom objektu u određenom trenutku i na koji način to smije izvesti. Pristup nekom resursu, dakle, definira kakve se akcije i operacije smiju obavljati nad određenim objektima. Akcije, na primjer mogu biti čitanje datoteke, pisanje u datoteku, stvaranje ili brisanje objekata, povezivanje sa nekim drugim programom, itd. Kad se spominje kokntrola pristupa, uzimaju se u obzir četiri situacije:

- sprečavanje pristupa – potrebno je osigurati da jedan korisnik ne može nanijeti štetu i narušiti integritet podataka nekog durgog korisnika na istom računalnom sustavu.
- ograničavanje pristupa – potrebno je osigurati da subjekt nema pristup nekim ključnim resursima kojima bi trebao moći pristupiti samo, na primjer, administrator sustava. Odnosno neki program ili korisnik ne bi smio imati ovlasti koje mu omogućuju nanošenje štete sustavu, kao što je podmetanje virusa i ostalih zlonamjerno oblikovanih programa.
- dozvola pristupa – sustav treba dozvoliti prijenos određenih prava pristupa s jednog korisnika na drugog. Na primjer, dozvola pristupa više korisnika jednom dokumentu.
- oduzimanje prava pristupa – treba omogućiti oduzimanje već dodijeljenih prava pristupa.

Kontrola pristupa uključuje autentikaciju (dokazivanje identiteta) i autorizaciju (dozvoljavanje pristupa uz uvjet pozitivne autentikacije). U nastavku su objašnjeni najvažniji modeli kontrole pristupa i sigurnosti.

2.1. Model temeljen na mogućnostima

Izraz mogućnost uveden je šezdesetih godina dvadesetog stoljeća. Na primjer, neka postoji računalni sustav u kojem program, da bi pristupio određenom objektu, mora imati posebnu značku (eng. *token*). Značka (*token*) određuje (označava) objekt i pruža programu (subjektu) dovoljne ovlasti za izvođenje određenog skupa akcija, kao što su čitanje ili pisanje, nad tim objektom. Primjer opisuje osnovnu ideju mogućnosti, gdje se značka naziva mogućnost.

Mogućnost se može usporediti s ključevima na privjesku za ključeve. Na primjer, ključ za automobil otključava samo točno određeno vozilo (označava pojedini objekt), i osoba koja posjeduje ključ može izvesti neke akcije nad vozilom, kao što su otključavanje ili zaključavanje vozila. Osoba koja posjeduje ključeve, može ih, posuditi i nekome drugome pa će ta druga osoba imati mogućnost otključavanja ili zaključavanja automobila. Dakle, ključ ne mora "znati" tko ga posjeduje, dovoljno je da ga netko ima. Na jednak način kao i ključ u danom primjeru, mogućnosti ne mare tko ih koristi.

Moguće su i varijacije ključeva za automobil. Na primjer, dvije uobičajene vrste ključeva su ključ za otključavanje i zaključavanje automobila (ne pokreće automobil) i ključ za pokretanje automobila (može otključavati, zaključavati i pokrenuti vozilo). Na jednak način, kao dvije vrste ključeva, dvije mogućnosti mogu označiti isti objekt (primjerice automobil) ali dati ovlasti za različite skupove akcija. Jedan program može imati samo mogućnost čitanja datoteka, dok drugi posjeduje mogućnost čitanja i pisanja u spomenutu datoteku. Kao u primjeru s ključevima, računalnim se resursima analogno dodjeljuju jedna ili više mogućnosti.

Mogućnosti se mogu delegirati. Ako jedna osoba da svoj ključ za automobil prijatelju, ona to čini na temelju povjerenja u prijatelja da on neće ključ proslijediti nekome drugome. Ukoliko ne postoji povjerenje u prijatelja, ne treba mu dati ključ.

Mogućnosti se mogu kopirati. Ako osoba koja posjeduje ključ posudi taj ključ svojem prijatelju, on može napraviti kopiju ključa. U praksi to nije problem jer osoba ne bi dala ključ prijatelju ako nema povjerenje u nj. U krajnjem slučaju ukoliko prijatelj iznevjeri osobu koja mu je dala ključ, osoba može promijeniti brave na automobilu. Slična se stvar može postići i sa mogućnostima i ta akcija je poznata kao odvajanje (raskinuće) objekta. Poništavaju se sve mogućnosti koje označavaju spomenuti objekt. Opozvana mogućnost nema nikakvih ovlasti.

Mogućnosti su jednostavne i u svakodnevnoj su upotrebi. Zaštita od krivotvorenja može biti ostvarena programski ili sklopovski. Programski pristup je pristupačniji jer se može ostvariti na bilo kojem računalu.

2.2. Liste kontrole pristupa (ACL)

Lista kontrole pristupa je uređeni skup podataka o ovlastima ili pravima pristupa svih subjekata (korisnika ili programa) na računalnom sustavu. Svaki korisnik ili grupa korisnika ima određena prava pristupa i ovlasti nad specifičnim objektom (direktorij ili datoteka). Svaki korisnik (subjekt) ima određena prava pristupa, kao što su ovlasti čitanja, pisanja ili izvođenja nad nekim objektima.
ACL se često koristi u okružjima gdje korisnici upravljaju svojom računalnom sigurnošću, kao što su sustavi Unix i Linux, a uobičajeni su za sveučilišta i istraživačke laboratorije. Prikladni su za područja primjene gdje je sigurnosna zaštita orijentirana na podatke, a sigurnosna politika je centralizirana. U sustavima koji imaju velik broj korisnika i gdje se stalno mijenjaju korisnici ili oni delegiraju prava pristupa nekom drugom subjektu na određeno vrijeme, nije preporučljivo koristiti liste kontrole pristupa.
U uobičajenom modelu liste kontrole pristupa svaki element liste određuje subjekt i operacije koje on smije obavljati nad određenim objektom. Na primjer unos (Alice, delete) u listi za neku datoteku daje Alice ovlasti brisanja te datoteke. U ACL modelu kada subjekt zatraži izvođenje neke operacije nad određenim objektima, sustav prvo provjerava listu i utvrđuje postoji li primjenjiv unos kako bi odlučio može li subjekt nastaviti sa izvođenjem operacije ili ne.
Dva su zanimljiva primjera listi kontrole pristupa, a to su implementacije na operacijskim sustavima Unix/Linux i Windows. Na operacijskim sustavima Unix/Linux, datoteke nemaju proizvoljan pristup listama, nego im se dodjeljuju atributi rwx (*r* označava dozvolu čitanja, *w* dozvolu pisanja, a *x* dozvolu izvođenja) za skupine vlasnik (eng. *owner*), grupa (eng. *group*) i ostali korisnici (eng. *others*). Atributi određuju ima li određeni korisnik prava čitanja, pisanja ili izvođenja nad nekim objektima. Direktorij sa svim zastavicama izgledao bi ovako:

```
drwxrwxrwx Alice Uprava
```

a zapis sa limitiranim pravima, primjerice, ovako:

```
-rw-r----- Alice Uprava
```

U primjeru je prikazana datoteka (prvi znak je -) čiji vlasnik (*owner*) ima pravo pisanja i čitanja nad njime, članovi grupe (*group*) imaju samo pravo čitanja, a svi ostali nemaju nikakvo pravo pristupa tom objektu. Vlasnik objekta je Alice, a grupa je Uprava. ACL na Linux-u sadrži samo korisnike, ne i programe tako da ne postoji način da se implementira uređena trojka (korisnik, program, objekt) kojoj se dodjeljuju prava pristupa. Na Linux sustavima postoji neizravna metoda rješavanja opisanog problema, a to je upotreba atributa SUID i SGID (U poglavlju s opisom primjera opisano je značenje ovih atributa).
Na operacijskim sustavu Windows situacija je slična, no umjesto atributa *read, write i execute* postoje odvojeni atributi *take ownership, change permissions i delete*, koji omogućuju fleksibilniju delegaciju prava pristupa.
Diskrecijski i mandatorni modeli (opisani u nastavku) mogu koristiti liste kontrole pristupa u svojim implementacijama.

2.3. Diskrecijski model

Diskrecijski model kontrole pristupa (eng. *Discretional model – DAC*) je oblik kontrole pristupa računalnim resursima (npr. direktoriji i datoteke) u kojem se pojedinim korisnicima dodjeljuje jedinstven tip pristupa, odnosno ovlasti. Na primjer, Bob može čitati i pisati u neku datoteku, a Alice ima samo ovlasti čitanja iste datoteke. Politiku (točno definirani mehanizmi i pravila sigurnosne zaštite) pristupa u ovom modelu određuje vlasnik objekta (računalnog resursa) gdje on definira tko može pristupiti tom objektu i koje ovlasti ima nad njime. Dva su važna koncepta kod ovog modela:

- **vlasništvo datoteka i podataka** – svaki objekt u sustavu ima svog vlasnika. U većini DAC sustava pretpostavljeni vlasnik objekta je onaj subjekt (korisnik ili proces) koji ga je stvorio. Politiku pristupa, dakle, određuje vlasnik objekta.
- **prava pristupa i ovlasti** – vlasnik objekta ih može dodijeliti ostalim subjektima na računalu za taj objekt

Korisnik kao vlasnik određenih datoteka i programa ima potpun nadzor nad njima, a uz to i nad dodjelom prava pristupa ostalim korisnicima. Prava pristupa se dodjeljuju samo onim subjektima kojima je to potrebno te se zbog toga model opisuje kao *need-to-know* model pristupa.
Model se temelji na prepoznavanju identiteta korisnika te pravilima pristupa koja se aktiviraju identifikacijom. Diskrecijski pristup objektima jedan je od najuobičajenijih modela na osobnim i umreženim računalima. Operacijski sustavi kao što su Windows te Unix/Linux uglavnom koriste DAC. Za razliku od nediskrecijskog pristupa, DAC ne ograničava kopiranje podataka, već ih prepušta korisnicima (subjektima). Ukoliko subjekt ima ovlasti čitanja, on može i kopirati podatke ako to želi. Ovo je bitno zbog toga što subjekt tada ima mogućnost pohraniti (kopirati) podatke u datoteku čiji je on vlasnik te dodijeliti ovlasti nad tim objektom proizvoljnim korisnicima, a to znači i korisnicima kojima je pristup izvornim podacima bio zabranjen. Na opisani način narušava se integritet podataka. U mandatornom modelu nije moguća ovakva situacija jer je zabranjeno kopiranje resursa na opisani način.
DAC modeli mogu biti temeljeni na strukturi vlasnik-objekt, matricama pristupa, centraliziranom, decentraliziranom ili raspodijeljenom pristupu. Često se implementira pomoću lista kontrole pristupa (ACL). Prema listi se određuju korisnici koji imaju pristup određenim objektima te koji tip pristupa i ovlasti im je dodijeljen (čitanje, pisanje ili izvršavanje). ACL ne pruža zaštitu od zlonamjerno oblikovanih programa kao što su trojanski konji koji se obično pokreću s ovlastima korisnika prijavljenog na sustav. Ukoliko subjekt pokrene neki od takvih programa (trojanski konji, virusi, itd.), tada ti programi mogu pristupiti objektima kojima može i prijavljeni korisnik. Alternativa listama kontrole pristupa u implementaciji DAC mehanizma su liste mogućnosti (eng. *capability lists*) za svakog korisnika. Takve liste određuju kojim resursima korisnik može pristupiti, a razlika od lista kontrole pristupa je ta što se ACL dodjeljuje objektu, a lista mogućnosti dodjeljuje se korisniku.
Za primjer je dan model sa matricama pristupa. Stupci predstavljaju datoteke, a redci korisnike. Neka su korisnici John, Alice i Bob. U pojednostavljenom primjeru knjigovodstva neka je John administrator, Alice upraviteljica i Bob zaposlenik. Neka su objekti kojima se pristupa operacijski sustav, program za knjigovodstvo, podaci te zapis kontrole knjiženja. John kao administrator ima sve ovlasti nad svim objektima, osim kontrole knjiženja (niti on ne bi smio imati pristup), Alice kao upraviteljica treba imati mogućnost pokretanja aplikacije na operacijskom sustavu, ali samo kroz odobrena sučelja. Tj., ona mora pisati i čitati podatke, a Bob kao zaposlenik ima samo mogućnost čitanja. Sljedeća slika prikazuje tablicu (matricu) ovlasti:

	Operacijski sustav	Program za knjigovodstvo	Podaci o računima	Kontrola knjiženja
John	rwx	rwx	rw	r
Alice	x	x	rw	-
Bob	rx	r	r	r

Tablica 1. Matrica ovlasti za pojednostavljeni primjer knjigovodstva

U tablici su upotrebljene oznake *r* za dozvolu čitanja, *w* za dozvolu pisanja, *x* za dozvolu izvođenja i – za potpunu zabranu pristupa.
U mnogim slučajevima primjer iz tablice 1 je dovoljno dobar, no za poseban slučaj knjigovodstva tablicu treba dopuniti.
Treba osigurati da Alice nema neograničenu dozvolu pisanja u datoteku s podacima o računima zbog opasnosti od pronevjere. Zbog toga se svako pisanje u spomenutu datoteku obavlja preko programa za knjigovodstvo. Također, John iz istog razloga kao i Alice ne smije pisati u datoteku s podacima. Sada dodijeljene ovlasti izgledaju kao u tablici 2:

	Operacijski sustav	Program za knjigovodstvo	Podaci o računima	Kontrola knjiženja
John	rwx	rwx	r	r
Alice	rx	x	-	-
Program za knjigovodstvo	rx	r	rw	w
Bob	rx	r	r	r

Tablica 2. Matrica ovlasti za poboljšani primjer knjigovodstva

Matrice kontrole pristupa su dobre za modeliranje sustava, kao i za implementaciju, no kod velikog broja korisnika javlja se problem zauzeća prostora i skaliranja. Na primjer banka sa 50 000 zaposlenika i 300 aplikacija imala bi matricu s 15 milijuna zapisa. Takva matrica je vrlo neprikladna te ne predstavlja samo problem u performansama računalnog sustava, već i sigurnosni problem u smislu ranjivosti na pogreške administratora. Kompaktniji način implementacije ovakvih informacija nudi model grupa i uloga.

2.4. Mandatorni model

Mandatorni model je oblik kontrole pristupa kod kojeg operacijski sustav ograničava mogućnost subjekta u pristupanju i/ili obavljanju neke akcije nad objektima. Subjektima i objektima je dodijeljena skupina sigurnosnih oznaka ili klasifikacija. Prisup određenim resursima dozvoljen je samo subjektima s dodjeljenom klasifikacijom. Mehanizme nadzora pristupa određenim objektima nameće operacijski sustav i/ili modul za sigurnost jezgre operacijskog sustava. Kad god korisnik ili program pokušaju pristupiti objektu, jezgra operacijskog sustava nameće autorizacijsko pravilo u ovisnosti o klasifikaciji te odlučuje ima li subjekt pravo pristupa. Svaka se operacija, koju korisnik želi izvesti nad nekim objektom, testira prema skupini autorizacijskih pravila, odnosno sigurnosnoj politici kako bi se utvrdilo je li operacija dozvoljena ili nije. U slučaju mandatornog modela sigurnosna politika se nadzire centralizirano te postoji vlasnik sustava koji nameće sigurnosnu politiku i sigurnosni administrator koji ju implementira. Korisnici ne mogu zaobići implementiranu politiku i, na primjer, pristupiti datotekama koje su im inače zabranjene. Za razliku od diskrecijskog sustava, koji također upravlja pravima subjekata, mandatorni model ne dozvoljava korisnicima odluku o sigurnosnoj politici i/ili dodjeljivanju sigurnosnih atributa. Administratori mogu implementirati sigurnosnu politiku koja se odnosi na cijelu organizaciju ili tvrtku. Korisnici ne mogu niti namjerno niti slučajno zaobići ili izmijeniti tu politiku. Na taj je način administratorima zajamčena mogućnost definiranja središnje politike koja se nameće svim korisnicima.

Mandatorni se model pristupa koristi u višerazinskim sustavima u kojima se obrađuju vrlo osjetljivi podaci, kao što su povjerljive informacije vladinih i vojnih organizacija. Višerazinski sustav čini računalni sustav koji upravlja sa više klasifikacijskih razina između subjekata i objekata. U mandatornim sustavima svi subjekti i objekti moraju imati oznake osjetljivosti. One definiraju razinu povjerenja potrebnu za pristup objektu. Kako bi subjekt pristupio danom objektu, on mora imati razinu osjetljivosti jednaku ili višu od one objekta kojem pristupa.

Vrlo važna funkcionalnost mandatornih modela je nadzor nad predajom i primanjem informacija od drugih sustava. Ova funkcionalnost mora osigurati ispravno održavanje i implementaciju oznaka osjetljivosti zbog ostvarenja prikladne sigurnosne zaštite.

Još od vremena Drugog svjetskog rata i Hladnog rata, zemlje NATO pakta uspostavile su sustav označavanja osjetljivosti dokumenata, tzv. klasifikacijske oznake koje imaju hijerarhijsku strukturu. Tablica 3 pokazuje tipičan primjer jedne takve strukture:

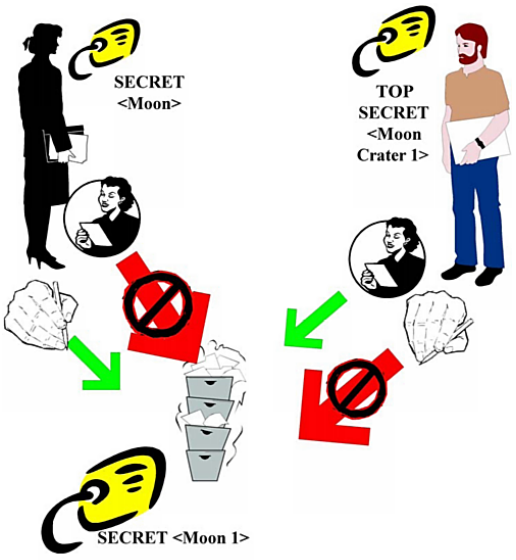
TOP SECRET
SECRET
CONFIDENTIAL
OPEN

Tablica 3. Hijerarhija klasifikacijskih oznaka

Izvorna je ideja bila da se informacije koje mogu stajati života označe kao *Secret*, a informacije koje mogu stajati mnogo života označe *Top secret*. Vladini zaposlenici imaju ovlasti u ovisnosti o dodijeljenom povjerenju. Politika kontrole pristupa je jednostavna, zaposlenik može čitati dokument samo ako ima minimalne ili veće ovlasti za tu akciju. Tako na primjer zaposlenik koji ima ovlasti *Top secret* može čitati dokument sa oznakom *Secret*, ali obratno ne vrijedi. Mandatorni višerazinski model analogan je opisanoj situaciji.
Postoje dvije poznate metode koje uključuju mandatornu dodjelu prava pristupa, a to su:

- Lattice
- Bell-LaPadula

Kod modela Lattice, svakom se objektu i subjektu dodjeljuje uređeni skup klasifikacija, kao što su one u tablici 3. Resursima s određenom klasifikacijom mogu pristupiti samo oni subjekti čija je klasifikacija viša ili jednaka onoj koju posjeduje spomenuti resurs. Lattice model se uglavnom koristi u vladine svrhe, a rijeđe u komercijalne svrhe.
Drugi model, Bell-LaPadula, kao i Lattice model sprečava pristup objektima s klasifikacijom manjom od klasifikacije subjekta (na primjer, neki proces sa određenom klasifikacijom ne može čitati podatke koji imaju višu klasifikaciju od njegove). Uz to, onemogućava se pisanje u objekte sa nižom klasifikacijom od subjekta koji želi obaviti akciju pisanja. Model uspoređuje oznaku objekta sa ovlastima korisnika i prema tome omogućuje ili brani pristup objektu. Sljedeća slika prikazuje kako djeluje ovaj model:



Slika 1. Bel-LaPadula model koristi i hijerarhijsku klasifikaciju i ovlasti subjekata

Na slici 1 moguće je uočiti da osoba s ovlastima *Secret* može pisati u dokument, ali ga ne može čitati, a osoba sa ovlastima *Top secret* može čitati, ali ne smije pisati. Lattice model za razliku od Bel-LaPadula modela dozvoljava pisanje korisniku sa *Top Secret* ovlastima.
Mandatorni model kontrole pristupa je usko vezan uz organizaciju sigurnosti u više razina (multi-level secure (MLS) systems). Poznata je tzv. Narančasta knjiga (eng. *Orange book* = *Trusted Computer System Evaluation Criteria* (TCSEC)) koju je izdalo Ministarstvo obrane SAD-a i u kojoj se definira mandatorni model kao sredstvo za ograničavanje pristupa objektima temeljeno na osjetljivosti podataka u objektima i na formalnoj autorizaciji subjekata koji mogu pristupiti informacijama određene osjetljivosti. Rane implementacije mandatornog modela bile su usredotočene na višerazinsku sigurnost, a neke od njih su *HPUX BLS, Harris CS/SX, i SGI Trusted IRIX.*
U novije doba javlja se sigurnosna implementacija za operacijski sustav Linux pod nazivom SELinux (eng. *Security Enhanced Linux*) koja je ugrađena u jezgre inačica 2.6 pa nadalje). MAC je takvim smjerom razvoja dobio veći značaj te je evoluirao izvan višerazinskog oblika sigurnosti. Ova novija ostvarenja ukazuju da je ranije spomenuta definicija iz Narančaste knjige nedovoljna za općenitu upotrebu. Na primjer, novije implementacije administratoru omogućuju usredotočenje na sigurnost računalne mreže i obranu od zlonamjerno oblikovanih programa bez mnogobrojnih ograničenja višerazinske sigurnosti.

2.5. Model temeljen na ulogama

Model temeljen na ulogama (eng. *Role Based Access Control – RBAC*) je način ograničavanja pristupa autoriziranim korisnicima. To je novija alternativa diskrecijskom modelu i mandatornom modelu. RBAC model nudi mogućnost implementacije i MAC modela i DAC modela kontrole pristupa. Prije pojave RBAC modela, postojali su samo mandatorni i diskrecijski modeli koji su međusobno isključivi.

Na primjer unutar organizacije moguće je kreirati uloge prema različitim funkcijama, gdje se prava za izvođenje određenih operacija dodjeljuju po ulogama. Svaka skupina korisnika može imati određenu ulogu pa se tako na primjer zaposlenicima dodjeljuje jedna ili više uloga. Svaka uloga ima određene ovlasti te svi korisnici koji imaju određenu ulogu imaju i ovlasti te uloge.

Ukoliko se, primjerice, razmatra struktura velikih organizacija, tada je moguće uočiti da se većina zaposlenika svrstava u jednu ili više manjih kategorija. Banka može imati između četrdeset ili pedeset takvih kategorija: bankar, računovođa, upravitelj itd. Ostalih (kao što su sigurnosni upravitelj i upravitelj mjenjačnice), kojima treba individualno dodijeliti prava pristupa, ima možda nekoliko desetaka. Dakle, potrebno je imati nekoliko grupa i funkcijskih uloga, koje se dodjeljuju zaposlenicima. Potrebno je naglasiti razliku između grupa i uloga, gdje grupu definira skup određenih pravila, a ulogu uređeni skup prava pristupa koja se mogu, u nekom vremenskom periodu upotrebe određenog resursa, vezati uz pravila. Tipičan primjer uloge je zapovjednik straže na brodu. Može postojati točno jedan zapovjednik straže u nekom vremenskom razdoblju. Postupak smjene dužnosti je također točno definiran. Moguće je kombinirati grupe i uloge. Na primjer, jedna grupa uloga može biti "Zapovjednici straže svih brodova trenutno na moru". U primjeru banke, upravitelj Cambridge podružnice može dobiti prava pristupa ukoliko je član grupe "upravitelji" i preuzeti ulogu "trenutni upravitelj Cambridge podružnice". Grupa "upravitelji" može izražavati rang u organizaciji, a uloga "trenutni upravitelj" može uključivati i zaposlenika nižeg ranga koji trenutno obavlja dužnost upravitelja jer su svi koji su rangirani iznad njega bolesni, na primjer.

Kako se korisnicima ne dodjeljuju prava pojedinačno već preko uloga, pojednostavljuje se dodjela ovlasti, kao i dodavanje ili brisanje korisnika. Dok liste kontrole pristupa dozvoljavaju ili brane neka prava pristupa, kao što su prava pisanja ili čitanja, one ne specificiraju na koje načine se objekt može mijenjati. U RBAC sustavu se dodjeljuje značenje svakoj dozvoljenoj operaciji nad objektima. Tako na primjer u operacije "kreiranje kreditnog računa" ili "testiranje šećera u krvi" imaju određeno značenje za neki objekt.

Ukoliko se uvede koncept hijerarhije u model temeljen na ulogama, moguće je simulirati već spomenuti Lattice model. RBAC se može smatrati nadskupom od LBAC.

Kod definiranja RBAC modela uobičajeno je koristiti sljedeće oznake:

- S (eng. *Subject*) - korisnik ili program,
- R (eng. *Role*) - uloga, tj. posao ili naslov koji definira razinu autoriteta,
- P (eng. *Permissions*) - dozvola načina korištenja resursa,
- SE (eng. *Session*) - mapiranje koje uključuje S, R i/ili P,
- SA - dodjela subjektu,
- PA - dodjela prava, te
- RH - parcijalno uređena hijerarhija uloga, može se predstaviti oznakom "≥".

Subjekt može imati više uloga, uloga može imati više prava, pravo pristupa se može dodijeliti različitim ulogama. Ograničenje postavlja restriktivno pravilo na potencijalno nasljeđivanje prava pristupa različitih uloga te se na taj način može postići odvajanje dužnosti. Na primjer, ne bi se smjelo dozvoliti istoj osobi kreiranje korisničkog računa za nekoga drugoga korisnika te istovremeno obavljanje procedure autorizacije.

Upotreba RBAC modela široko je primjenjiva na svim vrstama računalnih sustava, onim jednokorisničim ili višekorisničkim. Sustavi kao što su Microsoft Active Directory, SELinux, FreeBSD, Solaris, Oracle DBMS, PostgreSQL 8.1, SAP R/3 i mnogi drugi koriste različite oblike RBAC modela. U organizaciji s više stotina sustava i aplikacija upotreba RBAC modela te njegovo održavanje postaje kompleksan problem bez hijerarhijskog kreiranja uloga i privilegija.

3. Primjeri na operacijskom sustavu Linux

Unix je operacijski sustav nastao iz pepela operacijskog sustava MULTICS OS krajem 60-tih godina prošlog stoljeća. Za razliku od propalog MULTICS sustava, koji je trebao biti sustav s poboljšanim sigurnosnim funkcionalnostima, Unix je nastao kao projekt iz hobija te nije imao nikakvih sigurnosnih mehanizama. Uskoro su operacijski sustav Unix te sustavi temeljeni na njemu poslali popularni višekorisnički sustavi te se javila potreba za sigurnosnim mehanizmima kao što je kontrola pristupa.
Na operacijskim sustavima temeljenim na sustavu Unix, kao što su Linux, Solaris, SunOS, AIX, HP-UX, FreeBSD, interakcija s korisnikom počinje upisom korisničkog imena i zaporke. Dakle svakog korisnika je potrebno autorizirati, što znači da sustav treba odlučiti hoće li dozvoliti korisniku pristup sustavu ili ne, odnosno pri autorizaciji se utvrđuje koje će akcije korisnik nakon prijave moći obavljati. Svakom se korisniku dodjeljuju određene ovlasti, kao što su na primjer čitanje ili pisanje određenih datoteka, pokretanje programa, itd. Uobičajena je dodjela prava na temelju korisničkog imena i zaporke. Na Linux sustavima postoji tzv. *superuser root* ili administrator koji ima identifikacijski broj, *uid*=0, te ima potpunu kontrolu nad svim računalnim resursima, kao što su datoteke, direktoriji, priključci, programi, memorija, itd. Ukoliko se radi o višekorisničkom sustavu, moguće je nekim korisnicima dozvoliti obavljanje administratorskih operacija na nekoliko načina:

- Dijeljenjem administratorske tj. *root* zaporke – najjednostavnije, no nije preporučljivo jer svi korisnici ne bi smjeli imati administratorske ovlasti.
- stvarnjem višestrukih administratorskih korisničkih računa – stvaranje više korisničkih računa s identifikacijskim brojem 0 (uid = 0) koji imaju različita korisnička imena i zaporke. Ova metoda također zbog sigurnosnih razloga nije preporučljiva.
- *sudo* naredba – u datoteci /etc/sudoers se nalazi popis naredbi koje smiju zadavati korisnici prijavljeni na sustav. Na primjer zapis:

```
/etc/sudoers:
smith myhost = (root) /usr/local/bin/mycommand
```

  znači da korisnik smith smije izvoditi naredbu */usr/local/bin/mycommand* na računalu *myhost* sa ovlastima administratora. Korisnik smith može pokretati program mycommand sa:

```
smith$ sudo -u root /usr/local/bin/mycommand
```

  Upotrebom sudo funkcionalnosti nije potrebno dijeljenje administratorske zaporke, već se svakom korisniku dodjeljuju naredbe koje smije izvršavati s povišenim ovlastima. Kod konfiguracije sudo naredbe potreban je oprez kako se korisnička zaporka ne bi pretvorila u administratorsku.

U sljedećim poglavljima opisani su primjeri MAC i DAC modela na operacijskim sustavima Linux te su opisane prednosti i nedostaci oba pristupa.

3.1. Primjer diskrecijskog modela

Kontrola pristupa datotekama u operacijskim je sustavima UNIX/Linux realizirana diskrecijskim modelom koji uključuje i upotrebu lista kontrole pristupa. Različiti sustavi temeljeni na operacijskom sustavu Unix koriste različite datotečne sustave (neki od njih su ufs, ext2, ext3, ResierFS), međutim svi oni implementiraju kontrolu pristupa datotekama na jednak način. U donjem je primjeru pokazano pokretanje naredbe za provjeru ovlasti nad datotekom usr/bin/id.

```
[bash]$ ls –l usr/bin/id
-rwxr-xr-x 1 root root 16996 2007-09-29 14:51 /usr/bin/id
```

U prvom retku ispisa nakon tzv. prompt oznake ([bash]$) nalazi se naredba za pregled prava pristupa datoteci usr/bun/id. Drugi redak predstavlja rezultat izvođenja. U njemu je moguće uočiti osam segmenata odvojenih razmakom, gdje posljednji segment označava naziv datoteke. Prvi, treći i četvrti

segmenti su bitni za pojam kontrole pristupa. U prvom segmentu nalaze se oznake prava pristupa u smislu čitanja, pisanja ili izvođenja datoteke. Dakle, od desetak znakova prvi je znak crtica (-) i ona označava je li riječ o datoteci, direktoriju, poveznici ili priključku (da se umjesto crtice nalazi znak d, to bi bio direktorij, l poveznica i s priključak). Ostalih devet znakova (tri puta po tri) označavaju prava pristupa za vlasnika, vlasničku grupu i sve ostale korisnike. Oznaka *r* znači da korisnik ima dozvolu čitanja, *w* da ima dozvolu pisanja, a *x* da ima pravo izvođenja. Prva tri znaka od devet spomenutih, a to su rwx predstavljaju ovlasti vlasnika (*owner*) datoteke, gdje on ima pravo čitanja, pisanja i izvođenja. Slijedeća tri znaka (r-x) su ovlasti grupe (*group*) dodijeljene datoteci te oni označavaju da vlasnička grupa ima pravo čitanja i izvođenja datoteke. Zadnja tri znaka (r-x) su ovlasti svih ostalih korisnika (*others*) i oni imaju prava čitanja i izvođenja datoteke. Treći i četvrti segment označavaju vlasnika datoteke, odnosno korisnika i grupu, respektivno. U ovom slučaju vlasnik datoteke je *root*, tj. administrator. Sljedeći primjer pokazuje sve mogućnosti dodjele prava:

```
d  rwxt rwx rwx
- type

   ---- owner

        --- group

             --- others
```

Kod datoteka su značenja oznaka intuitivna, ali kod direktorija se mogu javiti određene nejasnoće. Zastavica x (bit za izvođenje) kod direktorija označava da je moguće pristupati datotekama unutar direktorija, ali da nije moguće vidjeti popis datoteka koje se u njemu nalaze. Popis datoteka je moguće vidjeti ukoliko je postavljena zastavica *r* za čitanje.
Opisane ovlasti se postavljaju naredbom *chmod*. Naredba *chmod* može imati numeričke ili alfabetske parametre kojima se postavljaju ovlasti na određeni objekt. Kod numeričkog načina troznamenkastim brojem se označavaju prava pristupa. Obično je taj broj u oktalnom zapisu jer postoje tri skupine kojima se dodjeljuju prava, a svaka skupina može imati postavljene tri oznake (r, w ili x) gdje bit 0 znači zabranu, a 1 dozvolu. Slijedeći primjer pokazuje upotrebu naredbe chmod:

```
[bash]$ chmod 600 test.txt
[bash]$ chmod u=rw test.txt
```

Naredbe su ekvivalentne, a označavaju da su vlasniku datoteke dodijeljena prava čitanja i pisanja (600 u binarnom zapisu je 110 000 000), dok svi ostali nemaju nikakva prava nad tom datotekom. Za detaljne upute korištenja naredbe *chmod* preporučljivo je pogledati *man* stranice operacijskog sustava Linux.
Moguće je dodijeliti još dva atributa datotekama, a to su SUID i SGID. SUID označava da će se datoteke nakon pokretanja izvoditi sa ovlastima vlasnika te datoteke, a ne sa ovlastima korisnika koji ju je pokrenuo. SGID je sličan atribut i on se analogno odnosi na grupu dodijeljenu datoteci. Administratorske datoteke s atributom SUID smatraju se velikim sigurnosnim rizikom jer ih napadač može iskoristiti za preuzimanje administratorskih prava i pokretanje proizvoljnih naredbi.
Na nekim operacijskim sustavima Unix\Linux pretpostavljene postavke prava pristupa datotekama nisu postavljene u smislu pružanja optimalne sigurnosne zaštite. Drugim riječima, nekim datotekama mogu pristupati korisnici koji ne bi smjeli imati pristup. Povijesno gledano, situacija je postala toliko ozbiljna da se propust počeo smatrati sigurnosnom ranjivosti. Na primjer, ranije inačice operacijskog sustava SunOS dozvoljavale su pisanje svim korisnicima u datoteku logfiles. Spomenuta situacija omogućavala je napadačima brisanje svih tragova napada. Uz to, neki proizvođači izdaju programe s nepotrebno postavljenim atributom SUID za administratorske aplikacije, što značajno povećava sigurnosni rizik. Dakle, važno je pažljivo prilagoditi prava pristupa na operacijskom sustavu.

3.2. Primjer mandatornog modela

Primjer mandatornog modela je komponenta *Security Enanced Linux* (SELinux). SELinux čini skup sigurnosnih politika u obliku modula za jezgru operacijskog sustava. SELinux nije zasebna distribucija već skupina modula koji se mogu primijeniti na bilo koji Unix\Linux sustav. SELinux je stvoren prema specifikacijama agencije NSA (eng. *National Security Agency*).

Jezgra koja ima integrirani SELinux nameće mandatorni model kontrole pristupa koji ograničava korisničke programe i sustavske servise na minimalne moguće ovlasti. Ovakav pristup reducira ili potpuno uklanja mogućnost pretvaranja programa u sigurnosni rizik i nanošenje štete sustavu. Mehanizam SELinux modula djeluje nezavisno od uobičajene kontrole pristupa na sustavima Linux. Ne postoji koncept administratora (root korisnika) te ne postoje atributi SUID i SGID.

Sigurnost sustava bez SELinux modula ovisi o ispravnosti implementacije jezgre, ispravnosti aplikacija te njihovih konfiguracija. Problem ili pogreška u bilo kojem od nabrojanih dijelova sustava može kompromitirati sigurnost cijelog sustava. Nasuprot tome sigurnost nadograđenog sustava sa SELinux modulima ovisi prvenstveno o ispravnosti jezgre i konfiguraciji sigurnosnih politika. Komercijalno se SELinux javlja kao dio Red Hat Enterprise Linux (RHEL) distribucije, inačica 4 i novijih. Podržana sigurnosna politika u RHEL4 distribuciji nije maksimalno stroga te je postavljena tako da u najvećoj mogućoj mjeri olakša rad na sustavu. Osim RHEL distribucije, SELinux komponenta je integrirana u sustavima Debian, Fedora (od inačice 2 na dalje), Gentoo, Ubuntu te većini sustava sa jezgrom inačice 2.6 i više.

Također razvijena je i jezgra pod nazivom SMACK (eng. *Simplified Mandatory Access Control Kernel*) koja implementira mandatorni model pristupa upotrebom posebnih oznaka pridruženih zadacima i spremnicima podataka. SMACK zahtijeva minimalnu aplikacijsku podršku i minimalnu količinu konfiguracijskih podataka.

3.3. Usporedba DAC i MAC modela

Većina se operacijskih sustava temelji na DAC modelu koji nameće sigurnost preko vlasništva. Ukoliko je korisnik vlasnik objekta, ima pravo postavljati ovlasti čitanja, pisanja i izvođenja nad tim objektom. U ovom modelu korisnici postavljaju kontrolu pristupa kako njima odgovara. Vlasnik sustava nema potpuni nadzor nad pristupom. Međutim najveću zabrinutost kod Linux sustava izaziva administratorski korisnički račun koji ima potpunu dozvolu pristupa i upravljanja sustavom i njegovim procesima. Ukoliko napadač preuzme ovlasti administratora, tada ih može iskoristiti za potpuno preuzimanje kontrole na nad računalnim resursima osjetljivog računala.

Kako bi se povećala sigurnost sustava trebalo bi ukloniti potrebu za administratorskim računom i prebaciti moć nad sustavom s korisničkih računa na vlasnika sustava. Takav mehanizam implementira MAC model u kojem sigurnosnu politiku nameće upravo vlasnik sustava. Ovdje korisnici, nakon implementacije sigurnosnih politika, ne mogu promijeniti ili zaobići dodijeljena prava pristupa.

4. Zaključak

Kontrola pristupa je važan sigurnosni element u svakom računalnom sustavu. U prethodnim poglavljima opisani su neki od najpoznatijih i najčešće korištenih modela kontrole pristupa i sigurnosti. U svakom višekorisničkom sustavu važno je implementirati primjerene sigurnosne mehanizme. Objašnjeni modeli kontrole pristupa implementiraju osnovnu funkcionalnost dodjele ovlasti i dozvole pristupa računalnim resursima te ograničavaju štetu koju sustavu mogu slučajno ili namjerno nanijeti korisnici ili programi. Najzanimljiviji primjeri implementacije modela kontrole pristupa su oni na operacijskim sustavima Linux. Većina Linux sustava ima ugrađeni diskrecijski model, no sve se više javlja interes za mandatornim modelom. Operacijski sustavi poput sustava Unix/Linux i Windows veliki su i kompleksni te se u njima često javljaju sigurnosni propusti u smislu programerskih pogrešaka. Ovdje najveći sigurnosni problem predstavljaju upravo oni propusti koji napadaču omogućuju preuzimanje administratorskih ovlasti. Kao rješenje ovog problema u Linux sustavu se javlja SELinux komponenta koja implementira mandatorni model kontrole pristupa. Osim osnovnih modela (DAC i MAC) postoji još i model temeljen na grupama i ulogama u kojem se prava pristupa korisnicima dodjeljuju preko uloga. Ovaj model svoju primjenu nalazi u velikim i vrlo velikim organizacijama s nekoliko stotina ili čak tisuća korisnika.

5. Reference

[1] R.Anderson: *Security Engineerinng*, http://www.cl.cam.ac.uk/~rja14/book.html, Cambridge 2001.

[2] Access Control, http://www.certiguide.com/secplus/cg_sp_11AccessControl.htm, veljača 2008.

[3] D.J. Barrett, R.G. Byrnes, R. Silverman: *Linux Security Cookbook*, O' Reilly, lipanj 2003.

[4] A. Chuvakin, C. Peikari: *Security Warrior*, O'Reilly, siječanj 2004.

[5] N. Dhanjani, *Hacknotes – Linux and Unix Security Portable Reference*, McGraw-Hill, 2003.

[6] Access Control List, http://en.wikipedia.org/wiki/Access_control_list , veljača 2008.

[7] Securing Linux with Mandatory Access Controls, http://www.linux.com/feature/113941, veljača 2008.

[8] Discretionary access control (DAC), http://uw714doc.sco.com/en/SEC_admin/IS_DiscretionaryAccCntlDAC.html, veljača 2008.

[9] SMACK, http://lwn.net/Articles/243921/, veljača 2008.