BEZPEČNOSTNÍ MODEL ZALOŽENÝ NA ROLÍCH (RBAC) A JEHO REALIZACE V XML SECURITY

Dagmar Brechlerová
KIT PEF ČZU Praha, Dagmar.brechlerova@pef.czu.cz

ABSTRAKT:
Mezi dnes často používané bezpečnostní modely patří tzv. RBAC - Role Based Access Control model. Tento model je velmi vhodný pro organizace, kde přístup ke zdrojům je závislý ne na konkrétním uživateli, ale jeho pracovní pozici. Příspěvek se zabývá také realizací tohoto modelu v jazyku XACML.

KLÍČOVÁ SLOVA:
XACML, RBAC, BEZPEČNOSTNÍ MODEL

ABSTRACT:
RBAC security models are often used today. This model is very useful for organization, where is not so much important name of subject or particular subject but his / her job function (role). This paper is also about XACML implementation of this model.

KEY WORDS:
XACML, RBAC, SECURITY MODEL

1.Bezpečnostní modely

Při formulování bezpečnostní politiky musíme popsat entity touto politikou řízené a stanovit pravidla, která tuto politiku tvoří. Je nutné zodpovědět např. tyto otázky:

- jaká je bezpečnostní politika?
- jaká dávat práva, komu, kam?

Toto dělají bezpečnostní modely. Některé bezpečnostní modely zdůrazňují utajení dat, jiné integritu dat. Některé modely jsou statické, jiné dovolují dynamické změny v přístupových právech. Pravidla pro definování politiky určují bezpečnostní modely. Dá se říci, že bezpečnostní modely mohou pomoci při vybudování určité bezpečnostní politiky, jsou jejím konkrétním vyjádřením. **Úkolem bezpečnostního modelu je jasně a velmi přesně formulovat bezpečnostní politiku.** Dále nutno podotknout, že hodnocení bezpečnosti vyžaduje u některých vyšších stupňů bezpečnosti, např. dle Orange Book i dalších kritérií neformální a posléze i formální model. Modelů byla vyvinuta již celá řada. Některé modely byly v praxi implementovány třeba v poněkud pozměněné podobě, jiné jsou pouze teoretické.

Modely mohou být užitečné např. v těchto příkladech:

- testování částečné bezpečnostní politiky
- dokumentování politiky
- jako pomoc pro implementaci bezpečnosti
- rozhodnutí, zda implementace splňuje původní požadavky
- studium těchto modelů vede k lepšímu pochopení, jak a proč chránit systém

Subjekt je aktivní prvek modelu; u některých modelů je to člověk, jinde proces. Aktivní

subjekt se snaží o nějakou operaci přístupu k pasivnímu objektu, je zde bezpečnostní monitor (reference monitor), který mu buď přístup povolí nebo zakáže.

Objekt je pasivní prvek např. soubor, adresář, tiskárna, paměť atd. Např. proces může být z tohoto hlediska jednou subjekt a jednou objekt. To samé může být jak objekt tak subjekt, záleží na tom, zda je v aktivní či pasivní roli.

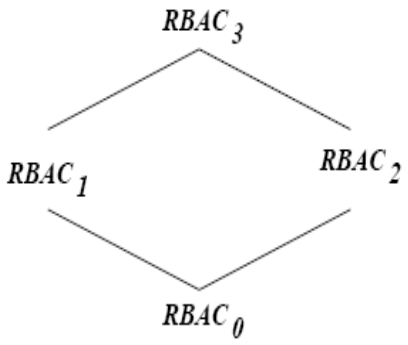
Jedním z dnes často užívaných modelů je Role Based Access Control Model - RBAC. V tomto modelu jsou uživatelé rozlišováni ne podle osobní identity, ale podle přidělené role. Např. v univerzitním prostředí se koncepce modelu RBAC jeví jako vhodná. Role může být např. student oboru Informatika 4. ročník, vyučující předmětu Databáze 1. Těchto vyučujících může být více, každý má stejná práva, jsou zastupitelní, při jakékoliv situaci (nemoc, odchod z pracoviště atd.) se pouze dosadí do role jiný konkrétní člověk. Určitá práva se mohou dědit. Např. vedoucí studijního oddělení má práva všech referentek, které jednotlivě mají práva pouze pro práci s daty studentů svého ročníku, oboru atd., za který odpovídají. Ale navíc má vedoucí nějaká další práva, která jednotlivé referentky nemají. Tento koncept je také velmi vhodný z hlediska ochrany soukromí. Pokud někdo žádá o určitý materiál, není nutné, aby se např. internetem přenášela informace, že uživatel Brechlerová žádal o to a to. Zcela postačí, pokud žádost zní, že vyučující předmětu Programování (tato informace je ověřena) žádá o přístup k materiálům a to pro akci zápis. To je v souladu s direktivou EU o ochraně soukromí a s celkovými trendy posledních let, kdy se o ochranu soukromí na Internetu začíná více dbát.

2. Role Based Access Control Model

Mezi hybridní politiky (a tedy modely) patří i RBAC politika (a model). Jedná se o model který vyvinul Sandhu a spol. [1], ve vývoji dále pokračovali David Ferraiolo a Richard Kuhn [2]. Tento model uživatele rozděluje podle jejich rolí (lékař, zdravotní sestra, ošetřující lékař, vyučující předmětu Programování 1 atd.). Jeden uživatel může mít různé role (vyučující předmětů Programovaní v jazyce C, Počítačové sítě, Databáze 1). Každá role je spojena s množinou povolených aktivit, které může uživatel v dané roli konat na zdroji, který je chráněn pomocí nadefinované bezpečnostní politiky. Množina povolených přístupů může být sdílena více rolemi.
Model se velmi hodí pro organizace, kde se mohou jednotliví lidé střídat v rolích (lékař, vyučující), ale kde role jsou poměrně trvalé. Nezáleží na konkrétní osobě, ale na roli, kterou osoba koná a pokud se třeba během semestru změní vyučující, tak pokračuje ve stejné roli. Zejména je tento systém velmi vhodný pro zdravotnictví, vzhledem k pevné hierarchii (sestra, lékař, primář, vrchní sestra atd.), ale i vysokoškolské prostředí má poměrně jasně definovanou hierarchii danou vyučujícími, cvičícími atd. Navíc podobný model je velmi flexibilní a jde upravit podle potřeb dané organizace.

Sandhu a spol.[1] popsali celkem 4 stupně RBAC modelů. RBACo, $RBAC_1$, $RBAC_2$, $RBAC_3$

$RBAC_o$ - má nejméně požadavků, role zde nemají žádnou hierarchii či strukturu. Je to základní model, jsou zde základní entity: uživatel (users U), role (R) a povolení (permission P). Vztahy mezi nimi jsou dále modulovány pomocí omezení (constrains). Dále je zde množina sessions (s). Uživatel je lidský uživatel: je možno model upravit tak, aby se jednalo i o další uživatele, nejen lidi, ale zde je konkrétně myšlen člověk. Role je pracovní funkce nebo pracovní název v organizaci s nějakou přidruženou informací týkající se autority a odpovědnosti v této funkci. Povolení P je mód přístupu k jednomu nebo více objektům v systému. Jde o autorizaci, přístupová práva nebo povolení (různá literatura používá různé názvy). Povolení je vždy kladné a vyjadřuje schopnost držitele povolení provést nějakou akci na systému.



Obr. Vztah mezi RBAC modely

$RBAC_1$ - množina rolí má hierarchickou strukturu, nadřazená role dědí povolené aktivity od podřízených rolí. Protože uspořádání nemusí být lineární, ale jde o částečné uspořádání, může nadřízená role dědit více možností (např. vedoucí katedry dědí práva vyučujícího Programování i práva vyučujícího Databáze, vedoucí studijního oddělení dědí práva referentek pro všechny ročníky atd.)
$RBAC_2$ používá omezení, která určují, zda různé varianty jednotlivých komponent modelů jsou povoleny (např. separace povinností, mohou být různé podmnožiny rolí, kde uživatel může být nucen nebýt členem více než jedné role).
$RBAC_3$ kombinuje předchozí dvě a tím pádem i RBACo a dále přidává určitá omezení. Modelům 1, 2, 3 se říká pokročilé modely.

RBAC model formalizovali Ferraiolo a Kuhn. [2]. Ti kromě subjektu a objektu ještě zavádějí tzv. Transakce. Ve své práci zavedli aktivní roli subjektu AR (s:subjekt)- role, kterou subjekt právě vykonává. Autorizovanou roli subjektu RA (s: subjekt) – každý subjekt může být autorizován pro jednu či více rolí. TA (r:role) – transakce autorizovaná pro roli, každá role může být autorizována pro vykonávání jedné či více transakcí. Subjekt může vykonávat transakci, exec(s,t) je pravda, jestliže subjekt s může vykonávat transakci t současné době, jinak to je nepravda.

Platí následující tři pravidla:

1. Přiřazení roli: Subjekt může vykonat transakci pouze, pokud subjekt má vybránu nebo přiřazenu roli:

   $$\forall s : subject, t : tran(exec(s,t) \Rightarrow AR(s) \neq \emptyset).$$

   Identifikace a autentizační proces (např. login) není považován za transakci. Všechny ostatní uživatelské aktivity na systému jsou vedeny skrze transakce. Takže každý aktivní uživatel musí mít nějakou aktivní roli.

2. Autorizace roli: Aktivní role subjektu musí být autorizována pro subjekt:

   $$\forall s : subject(AR(s) \subseteq RA(s)).$$

   Spolu s (1), toto pravidlo zajišťuje, že uživatel může vykonávat pouze ty role, pro které je autorizován.

3. Autorizace transakce: subjekt může vykonávat transakci pouze jestliže transakce je autorizována pro aktivní roli subjektu.

   $$\forall s : subject, t : tran(exec(s,t) \Rightarrow t \in TA(AR(s))).$$

Role mohou být členy dalších rolí. Dále jsou v této základní práci probrány důsledky výše uvedených pravidel.

Z výše uvedených prací vychází standard pro RBAC [3], který ještě přidává možnost uživatelských sezení, což dovoluje v rámci tzv. sezení role aktivovat nebo deaktivovat.

Základní vlastnosti jsou shrnuty v Core RBAC. Hierarchical RBAC definuje hierarchii rolí a Constrained RBAC definuje statické a dynamické oddělení vztahu pravomocí.

3. XACML a RBAC

V jazyku XACML [5],[6],[7],[8],[9] pro tento model existuje speciální profil, **RBAC XACML profil.**

Pro řešení některých problémů spojených právě s RBAC (role- based access control) v XACML verze 2.0 (i dříve) existuje speciální profil, RBAC profil, plnou specifikaci je možno najít v materiálu [4]. V tomto profilu je zahrnut **Core RBAC** a **Hierarchický RBAC**. Nejsou nutné žádné změny v XACML verze 1.0, 1.1, 2.0, je to rozšíření RBAC verze 1.0, navíc je zaveden nový pojem (prvek) v XACML, a to AttributeId for roles. Následuje základní popis tohoto profilu a jeho použití je ilustrováno na návrhu pro univerzitní praxi.

Základní pojmy používané v tomto profilu jsou následující:

Atribut - odvolává se na XACML <attribute>. Má jméno, identifikátor dat a hodnotu, každý atribut je spojen buď se subjektem, chráněným zdrojem, nebo akcí, která má být prováděna na zdroji, nebo prostředí. Atributy v politice jsou buď selektor, nebo designator.

<AttributeSelector> (XPathvýraz) nebo <SubjectAttributeDesignator>,
<ResourceAttributeDesignator>,
<ActionAttributeDesignator>, a <EnvironmentAttributeDesignator>.

Junior role - Role A je junior Roli B, pokud dědí veškerá povolení Role B.
Multi role povolení: množina povolení, pro které uživatel musí najednou plnit více rolí, např. musí být vyučující více předmětů, musí být zároveň pedagog i správce sítě apod.
Permission - povolení nějaké akce na nějakých zdrojích, za určitých podmínek.
PPS permission Policy set.
Role Enablement Authority – entita, která přiřazuje role.
RPS Role Policy Set.
Senior role je nadřazená role. A je senior vůči B, pokud má A veškerá práva B.
Politika je množina pravidel. RBAC; politika je specifikována tak, že subjekty jsou role, spíše než individuální jednotlivci.

Politiky specifikované v tomto profilu dovedou odpovědět tři typy otázek:

Pokud má subjekt role R_1, R_2, R_3..R_n, může provozovat na daném zdroji dané akce?
Je subjektu X povoleno mít roli Ri?
Jestliže subjekt má role R_1,R_2,..R_n povoleny, znamená to, že má subjekt povolení spojená s danou rolí R_m? Tj. je role R_m buď rovna nebo junior k některé z rolí R_1, R_2, R_3. nebo...R_n, ?

Role
Role jsou určeny jako XACML subjekt atributy, pouze jsou dvě výjimky.
Role Assignment <Policy Set> a HasPrivilegiesOfRole <Policy >, zda jsou role Resource atribute.

Možnosti tohoto profilu neobsahují zodpovězení otázky typu: **jakou množinu rolí má subjekt X**? To je ponecháno na tzv. **Role Enablement Authority**, jejíž implementace ale není specifikována a která není obsažena v XACML PDP. Role je zde určena v době autorizačního rozhodnutí, v této verzi není možné určovat role dynamicky, to snad bude umět další verze. Za předpokladu, že role subjektu musejí být určeny resp. musejí být určeny ve chvíli vyhodnocení politiky, profil nabízí zformování dvou speciálních typů Policy sets, aby popsaly povolení pro různé role. Tyto jsou zvány **Role Policy Set (RPS)** a **Permission Policy Set (PPS)**. RPS je aplikovatelná na speciální roli. Aktuální povolení pro roli jsou specifikována v PPS, na kterou je odkaz v RPS. Tento způsob umožňuje dědit veškerá práva jiné role tím, že je zahrnut odkaz na junior PPS v senior PPS. Viz obrázek dále.

HasPrivilegesOfRole policy je volitelný typ politiky, která je pak zahrnuta v PPS a dovoluje dávat dotaz: má daný subjekt privilegia určité role?

Jak je implementován RBAC model do XACML?

Core RBAC obsahuje základních 5 datových elementů:

- **users**
- **roles**
- **objects**
- **operations**
- **permissions.**

Dále je popsána implementace těchto prvků do XACML:

User to je XACML Subject, mohou být užity všechny Subject Category hodnoty.
Role je XACML Subject Attributes, množina rolí je charakteristická pro danou aplikaci a danou doménu, různá použití role mohou být různá, neexistují žádné standardní role, mohlo by dojít ke zmatení, ale je určité doporučení.
V každé aplikaci či doméně by měla být publikována jednotná množina hodnot, které budou aplikovány na roli, **a bude o tom udělána dohoda tj. množina Attributed values, Data Type values atd..**
Object je XACML Resource.
Operation je XACML Action.
Permission jsou XACML Role <Policy Set> a Permission<Policy Set>.

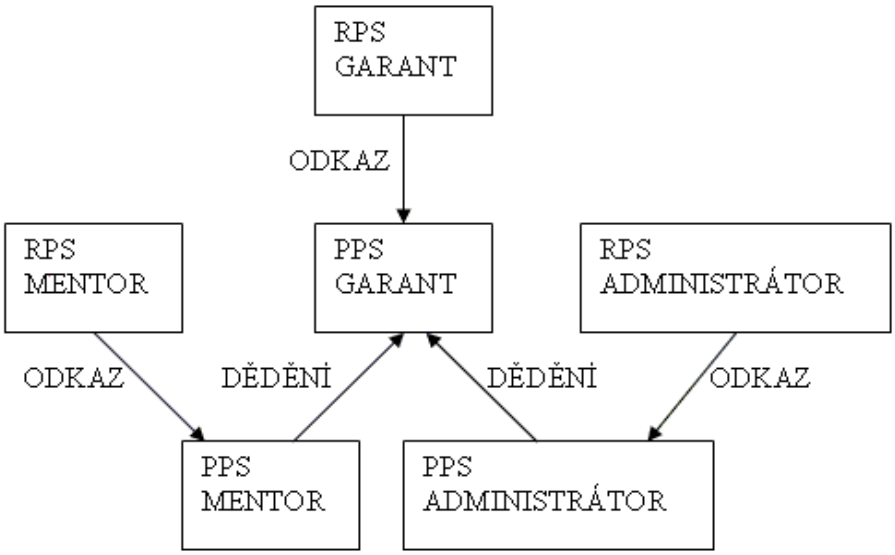
Core RBAC podporuje možnosti, že více uživatelů má stejnou roli, jeden uživatel má více rolí, role má více povolení, a více rolí má stejné povolení. Tj. např. v univerzitním prostředí více uživatelů má roli cvičícího předmětu Algoritmizace. Každý pedagog, ať vyučuje cokoliv, smí nahlížet do záznamu studenta. Jeden uživatel má více rolí, tj. např. určitý pedagog má roli vyučující Algoritmizace i Sítě atd. Všechny tyto podmínky umí XACML splnit. Pedagog smí zapisovat třeba známky a měnit materiály. **Ale XACML neumí přidělovat role uživatelům, to musí dělat jiný prvek**

Hierarchický RBAC navíc povoluje dědění.

XACML umí řešit některé problémy vyjádřené pomocí RBAC modelu, ale ne všechny. Některé je nutno vyjádřit nějak jinak za pomoci plné verze jazyka XACML.

Příklad na použití RBAC XACML v univerzitní praxi.

Mějme studijní materiál, **garant** tento materiál vytváří a odpovídá za něj, cvičící (**mentor**) ho smí pouze číst, nesmí do něj zasahovat. **Administrátor** materiálu smí dělat opravy v určité části textu, jiné ale nesmí ani číst. Garant je nadřazená role jak cvičícímu, tak administrátorovi. Garant potřebuje někdy povolit cvičícímu, aby také dělal příslušné úpravy, pro tuto situaci bude cvičící mít roli cvičícího i administrátora. Musí tedy v určité chvíli mít obě role. Existuje zde role Garant, role Mentor a role Administrator. Role Mentor a Administrator jsou junior role roli Garant. Mentor má právo pouze číst celý materiál, Administrator smí upravovat určité části, nesmí ale číst vše. Garant dědí obě tato práva a navíc tvoří celý materiál.



Obr. Vztahy mezi PPS a RPS

Jak již uvedeno dříve, XACML RBAC může volitelně podporovat dotaz "Má tento subjekt priviegia role X?" Pokud ano, pak PPS **musí** obsahovat HasPrivilegesOfRole<*Policy*>, je to volitelný typ politiky.

Přidělování různých atributů rolí jednotlivým uživatelům je mimo schopnost XACML. Musí zde být jedna či více entit, které odkazují na Role Enablement Authority, implementované tak, aby vykonávaly tyto funkce. Tento profil předpokládá, že přítomnost XACML Request Context atributů rolí pro daného uživatele je známo před řešením přístupu. Odkud mohou atributy rolí přijít? Je to implementačně závislé. Někdy mohou atributy rolí přicházet z identity management servisu, který udržuje informace o uživateli včetně pro uživatele přiřazené nebo povolené role. Potom tento identity management působí jako Role Enablement Authority. Statické role mohou být uloženy v LDAP adresáři a PDP Context Handler z XACML si je tam odsud získá. Další možností je použít DN (Distinguished Name) z X.509 certifikátu k popisu role uživatele. XACML má celou řadu funkcí X 500, které dovedou DN

použít. Problematické může být to, pokud se role uživatele mění, např. lékař, který pracuje v nemocnici potřebuje určitou roli, ale při výjezdu se záchrannou službou musí mít již jiná práva tj. jinou roli. Tj. Role uživatele se pak mění v závislosti na umístění uživatele. Tento problém je velmi palčivý zejména ve zdravotnictví, v univerzitním prostředí není tato otázka aktuální.

Závěr: RBAC a jeho implementace v jazyku XACML jsou velmi významným prvkem pro použití rolí v informačním systému.

LITERATURA

[1] R. S. Sandhu, E.J. Coyne, H.L. Feinstein, C.E. Youman (1996), "Role-Based Access Control Models", IEEE Computer 29(2): 38-47, IEEE Press,
[2] D.F. Ferraiolo and D.R. Kuhn (1992) "Role Based Access Control" 15th National Computer Security Conference
[3] http://csrc.nist.gov/groups/SNS/rbac/
[4] http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-rbac-profile1-spec-os.pdf
[5] Brechlerová, D. XML a bezpečnost II., Crypto-World. Roč. 9, č. 2 (2007),
[6] Brechlerová, D., RBAC Access Model and its Realization by XACML. Agrární perspektivy. Praha : Česká zemědělská univerzita, 2007.
[7] Brechlerová, D., XACML - Kontrola přístupu, Agrární perspektivy XIV. Znalostní ekonomika. Praha : Česká zemědělská univerzita, 2005.
[8] Brechlerová, D., XML Security in 2007, The Internet and Organisational Security. Zlín : Univerzita Tomáše Bati, 2007.
[9] http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml#XACML20