Access control policies modeling, representation and analysis

Clara Bertolissi

Aix-Marseille University, LIS UMR CNRS 7020, France

April 8, 2021

Outline

1. Overview on access control
2. A unified framework based on categorization : CBAC
   - Axiomatization
   - Operational Semantics
   - Graphical representation of CBAC policies
3. Conclusions

Introduction

" A logical component that serves to
i) receive the access for an Object from a Subject
ii) and decide and enforce the access decision "

a definition from NIST

Introduction

- Access control model :
  abstract representation of notions of relevance for access control.
  - principals, resources, privileges, actions on resources, ...
- Access control policies
  associate privileges with principals.
- Many models with increasing power have been defined in the last decades.

AC Models

Discretionary Access Control (DAC)

The access control is to the discretion of the object's owner.

The owner can determine who should have access rights to an object and what those rights should be.

		OBJECTS								
		subjects			files		processes		disk drives	
		S_1	S_2	S_3	F_1	F_1	P_1	P_2	D_1	D_2
SUBJECTS	S_1	control	owner	owner control	read *	read owner	wakeup	wakeup	seek	owner
	S_2		control		write *	execute			owner	seek *
	S_3			control		write	stop			

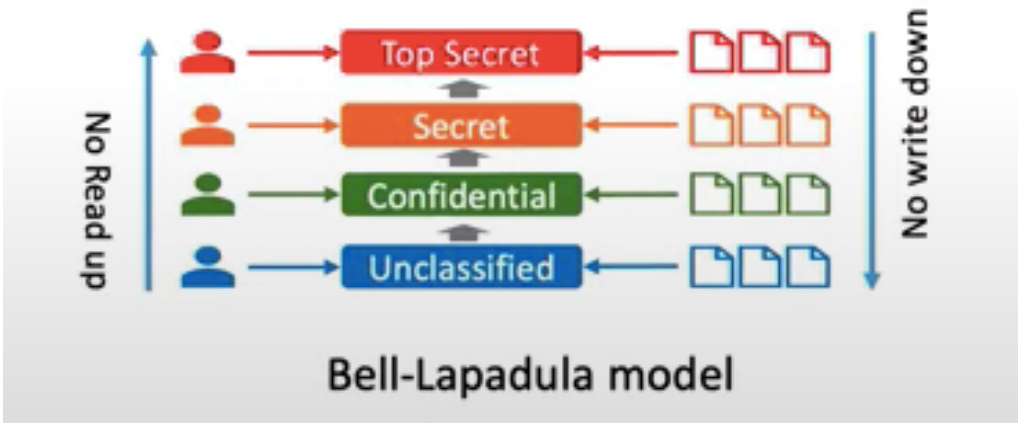
* - copy flag set

Access control matrix model

AC Models

Mandatory Access Control (MAC)

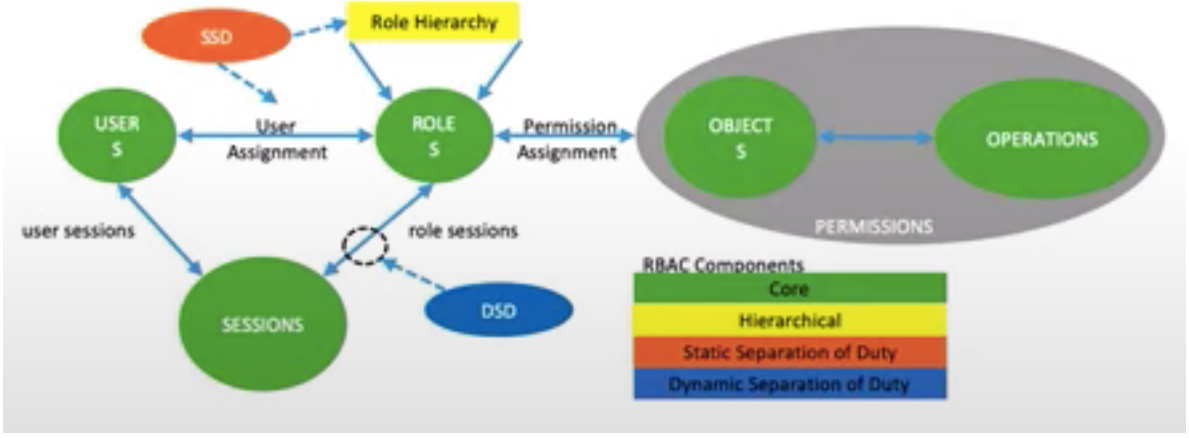
Access control policy decisions are made by a central authority, not by the individual owner of an object. User cannot change access rights.



Introduction

Role-based Access control [Ferraiolo,Kuhn'92]

- A user is assigned to privileges based on roles, reflecting the permissions needed to perform defined functions within an organization.
- Role permissions may be inherited through a role hierarchy.
- Many variants : time-based, location-based, etc. ...

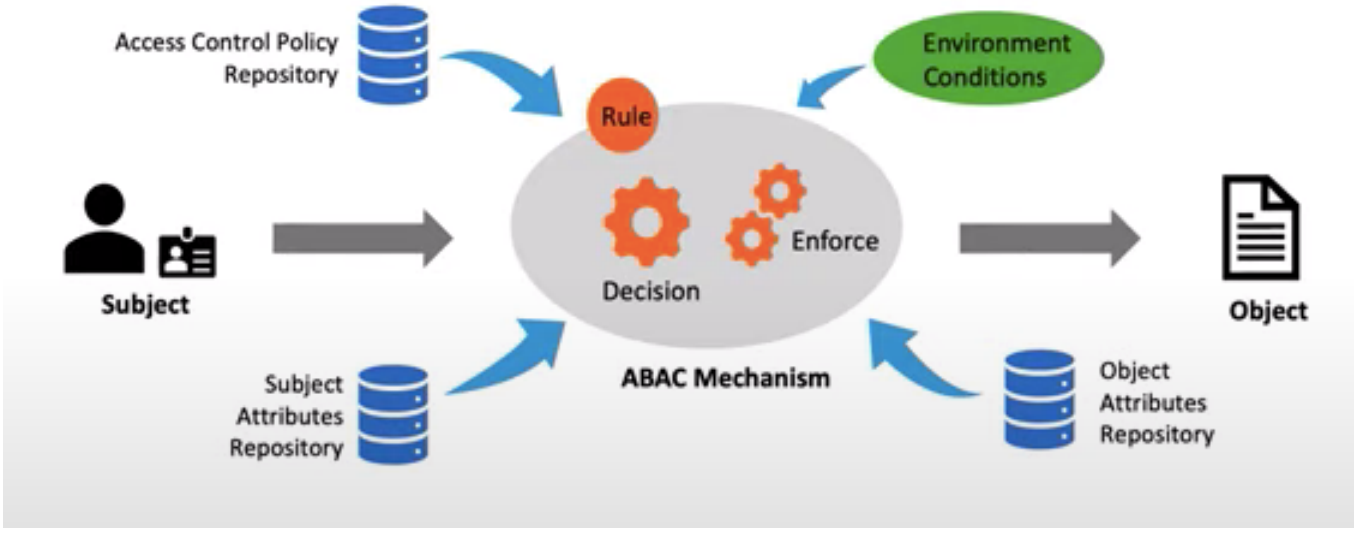


Introduction

Attribute-based Access Control (ABAC)

[Wang, Wijesekera,Jajodia'04]

- access is mediated based on attributes associated with subjects (requesters) and the objects to be accessed.
- An access control rule set defines the combination of attributes under which an access may take place.



Introduction

Idea : identifying the core concepts commun to the different existing models.

$\Rightarrow$ definition of a meta-model of access control [Barker'09]

$\Rightarrow$ unifying model with rule-based semantics: CBAC [Inf&Comp'14]

Formal approach :
entities, relationships, axioms for access control

Extensions with prohibitions, obligations, distributed semantics, ...

The CBAC model

i) a family $\mathcal{E}$ of sets of **entities**: principals $\mathcal{P}$, actions $\mathcal{A}$, resources $\mathcal{R}$, ... which are classified into **categories** $\mathcal{C}$ (seen as groups of entities).

ii) a family $\mathcal{R}el$ of **relationships between entities**,

- *Principal-Category Assignment*, $\mathcal{PCA}$: $(p, c) \in \mathcal{PCA}$
- *Resource-Category Assignment*, $\mathcal{RCA}$: $(r, c) \in \mathcal{RCA}$
- *Permissions*, $\mathcal{ARCA}$: $(a, c_r, c_p) \in \mathcal{ARCA}$
- *Authorisations*, $\mathcal{PAR}$: $(p, a, r) \in \mathcal{PAR}$

The CBAC model

iii) **axioms** that specify the properties the relationships should satisfy.

$$(ax) \quad \forall p \in \mathcal{P},\ \forall a \in \mathcal{A},\ \forall r \in \mathcal{R}, (\exists c_p \in \mathcal{C}, \exists c_r \in \mathcal{C}, (p, c_p) \in \mathcal{PCA} \wedge (r, c_r) \in \mathcal{RCA} \wedge (a, c_r, c_p) \in \mathcal{ARCA}) \Rightarrow (p, a, r) \in \mathcal{PAR}$$

- A reflexive-transitive relation $\subseteq$ may be added to model **category hierarchy** and be included in a generalised version of axiom (ax).

Example: bank policy

- The policy considers several principals and their functions (bank manager, teller, financial adviser, personal banker,...).
  $\mathcal{PCA} = \{$(John Smith, manager), (Bob Duval, bank teller),... $\}$
- There are two main resource kinds, accounts and investments, (that may be specialised into e.g. personal account or saving account)
  $\mathcal{RCA} = \{$(Lynns account, saving account), (McGregor insurance, investment),...$\}$
- Standard actions like open, close or read an account, and register, delete and validate an investment are defined.
  $\mathcal{ARCA} = \{$(bank manager, open account, saving account), (financial adv, register, investment),... $\}$
- Policies authorizations are derived using the axiom (ax1)
  (John Smith, open account, Lynns account) $\in \mathcal{PAR}$

Category-based access control model

Advantages of our formal meta-model:

- compare and compose policies rigorously [PPDP'08]
- better understand the consequences of changes [PPDP'16,TCS'17]
- develop analysis techniques (to deal with policy conflicts, to prove properties of policies,...) [STM'10,Inf&Comp'14]
- re-use work (in all instances of the generic model) e.g.
  - RBAC : role = category [DBSec'06]
  - ABAC : categories are used to specify and structure the relation between attributes and permissions [CODASPY'19]

Rewrite-based operational semantics

The operational specification of a CBAC policy can be defined by a set of rewrite rules :

$$
\begin{array}{lcl}
par(p,a,r) & \rightarrow & \textit{if } \; zip(a, rca(r)) \cap arca^*(pca(p)) \neq \emptyset \\
 & & \textit{then grant else deny} \\
pca(p) & \rightarrow & [c_1, \ldots, c_n] \\
rca(r) & \rightarrow & [c'_1, \ldots, c'_m] \\
arca(c_i) & \rightarrow & [(a_1, c'_1), \ldots, (a_n, c'_n)] \\
zip(a, \mathrm{nil}) & \rightarrow & \mathrm{nil} \\
zip(a, cons(c, l)) & \rightarrow & cons((a, c), zip(a, l)) \\
arca^*(\mathrm{nil}) & \rightarrow & \mathrm{nil} \\
arca^*(cons(c, l)) & \rightarrow & append(arca(c), arca^*(l)) \\
\ldots &
\end{array}
$$

Proposition

$par(p, a, r) \rightarrow^* grant$ if and only if $(p, a, r) \in \mathcal{PAR}$.

Rewrite-based specification

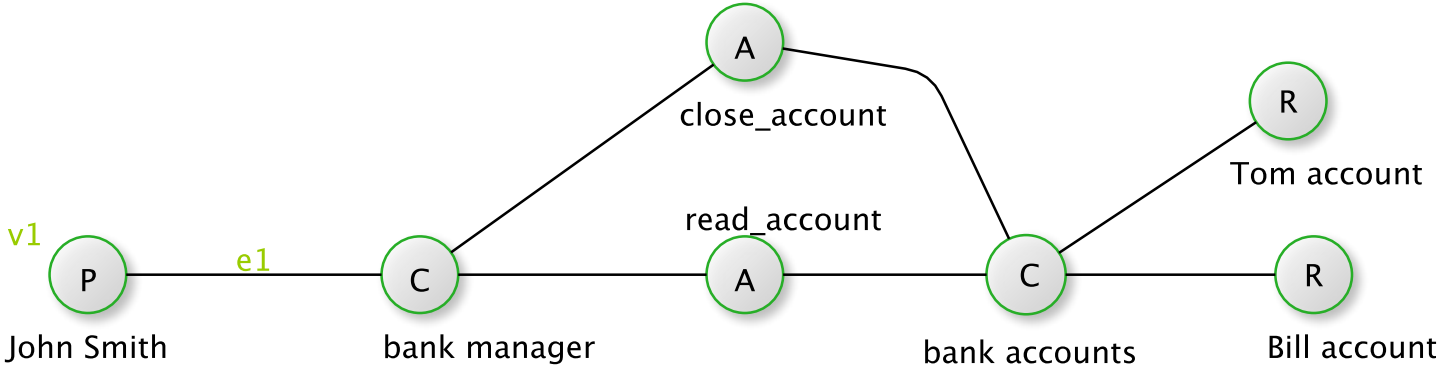
- make use of rewrite-based frameworks (such as CiME, MAUDE or TOM) to evaluate policy queries.

- perform automated policy analysis:
  - Consistency of a policy: every access request receives a unique answer.
  - Termination of a policy: every access request evaluation returns an answer.
  - Totality: every access request evaluation returns a decision.

  $\Rightarrow$ Properties of policies are checked using confluence and termination of sets of rewrite rules.
  Rich literature and automated tools like Aprove available.

Graphical representation for CBAC policies

An *policy graph* is a tuple $\mathcal{G} = (\mathcal{V}, E, lv, le)$. where nodes and edges are labelled by records.



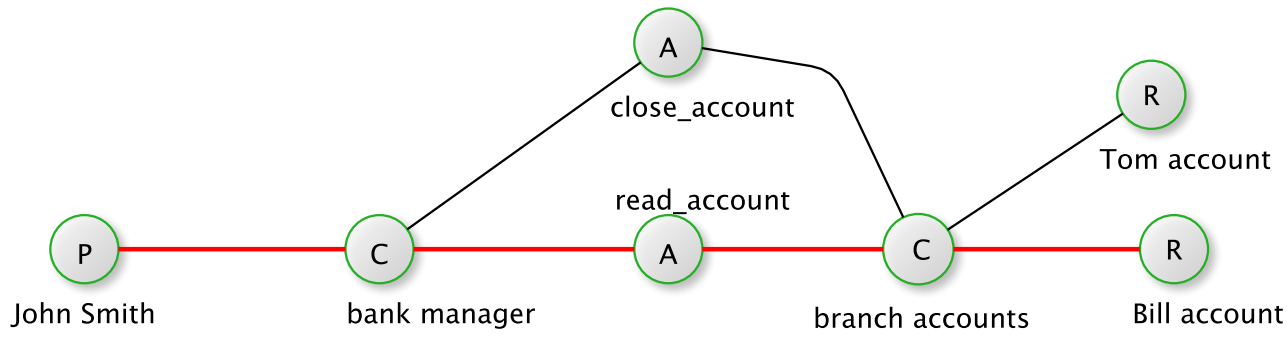
$lv(v_1)$= {ent="John Smith", type=P, age= 32, ...},
$le(e1)$= {adj=v_1 v_2, type=PC, ...}

A policy graph is *well-formed* if it satisfies certain type constraints and it has no redundant edges.

CBAC policies and policy graphs

Given a well-formed policy graph, we can extract

- entities $\mathcal{E}^{\mathcal{G}}$ as the nodes in the graph, and
- relations $\mathcal{R}el^{\mathcal{G}}$=$\{\mathcal{PCA}^{\mathcal{G}}, \mathcal{RCA}^{\mathcal{G}}, \mathcal{ARCA}^{\mathcal{G}}, \mathcal{PAR}^{\mathcal{G}}\}$



PAR relation

Proposition

Let $\mathcal{G} = (\mathcal{V}, E, lv, le)$ be a well-formed policy graph.
Then $\langle \mathcal{E}^{\mathcal{G}}, \mathcal{R}el^{\mathcal{G}} \rangle$ defines a CBAC policy.

Proposition

For any CBAC policy $\langle \mathcal{E}, \mathcal{R}el \rangle$ there exists a well-formed policy graph $\mathcal{G}$ such that $\mathcal{E} = \mathcal{E}^{\mathcal{G}}$ and $\mathcal{R}el = \mathcal{R}el^{\mathcal{G}}$.

Policy review queries

Policy review queries directly examine the content of policies.

- which principals are authorised to perform a given operation on a resource?
- for a given principal, what are the associated permissions?
- each user has at least one permission, each resource can be accessed by at least one user (Effectiveness)
- Are all the resources accessible (in terms of principals and permissions)? (Liveness)

$\Rightarrow$ can be checked in polynomial time in the size of the policy by graph traversal.

Extensions and applications

- Administrative CBAC model specified within CBAC
  [CODASPY'20,21]
- A Data Access Model for Cloud Storage
  [ICSS'18,SACMAT20]

Conclusions

- CBAC : a formal framework for access control policy specification
- formal operational rewrite-based semantics
- graph-based representation of CBAC policies,
- policy properties can be checked using rewrite techniques and graph traversal algorithms.

Future work:

- study policy composition and develop techniques to compare policies represented by graphs,
- use the graph modeling tool PORGY[1] to visualise and simulate CBAC and *Admin-CBAC* policies.

[1] `https://porgy.labri.fr`

Conclusions

Line of work done in collaboration with: *M. Fernandez, B.Thuraisingham, S. Barker, S. Alves, J. Jaimunk, and more... (see references)*

Thank you!

Any questions?

Selected references

- Clara Bertolissi, Maribel Fernández, Bhavani M. Thuraisingham: Graph-Based Specification of Admin-CBAC Policies. CODASPY 2021: to appear.
- Clara Bertolissi, Maribel Fernández, Bhavani M. Thuraisingham: Admin-CBAC: An Administration Model for Category-Based Access Control. CODASPY 2020: 73-84
- Maribel Fernández, Alex Franch Tapia, Jenjira Jaimunk, Manuel Martinez Chamorro, Bhavani M. Thuraisingham: A Data Access Model for Privacy-Preserving Cloud-IoT Architectures. SACMAT 2020: 191-202
- Maribel Fernández, Ian Mackie, Bhavani M. Thuraisingham: Specification and Analysis of ABAC Policies via the Category-based Metamodel. CODASPY 2019: 173-184
- Clara Bertolissi, Jean-Marc Talbot, Didier Villevalois: Analysis of access control policy updates through narrowing. PPDP 2016: 62-75

Selected references

- Sandra Alves, Maribel Fernández: A graph-based framework for the analysis of access control policies. Theor. Comput. Sci. 685: 3-22 (2017)
- Clara Bertolissi, Maribel Fernández: A metamodel of access control for distributed environments: Applications and properties. Inf. Comput. 238: 187-207 (2014)
- Clara Bertolissi, Worachet Uttha: Automated analysis of rule-based access control policies. PLPV 2013: 47-56
- Steve Barker, Clara Bertolissi, Maribel Fernández: Action Control by Term Rewriting. Electron. Notes Theor. Comput. Sci. 234: 19-36 (2009)
- Clara Bertolissi, Maribel Fernández, Steve Barker: Dynamic Event-Based Access Control as Term Rewriting. DBSec 2007: 195-210