Security Engineering
Fall 2015

Lecture 09 – Identity and Access Management

Fabio Massacci

Identity and Access Management

- ***Central element of computer security***
- ***ITU X.800 – Access Control Definition***
  - "The prevention of unauthorized use of a resource, including the prevention of use of a resource in an unauthorized manner''
- ***CSA – IAM definition***
  - includes people, processes, and systems that are used to manage access to enterprise resources by assuring that the identity of an entity is verified, then granting the correct level of access based on the protected resource, this assured identity, and other context information.

Identity and Access Management - II

- ***CSA Definition***
  - includes people, processes, and systems that are used to manage access to enterprise resources by assuring that the identity of an entity is verified, then granting the correct level of access based on the protected resource, this assured identity, and other context information.
- ***Who***
  - people, processes, and systems
- ***When***
  - used to manage access to enterprise resources
- ***What happens?***
  1. assuring that the identity of an entity is verified,
  2. granting the correct level of access based on
     - the protected resource,
     - this assured identity, and
     - other context information.

Component Number ZERO

- ***The POLICY***
  - A specification of what is a "correct level of access" and who are the "identities" for which this level is appropriate and the "contextual conditions"
  - In many case it should "formal"
    - not necessarily in the formal logic sense
- ***Often forgotten in the security analysis***
- ***``If a program has not been specified, it cannot be incorrect; it can only be surprising.''***
  - W.D. Young, W.E. Boebert, and R.Y. Kain, "Proving a Computer System Secure", Scientific Honeyweller, 6(2):18--27, July 1985
  - More precisely the "actual" system defines what is the "correct" level of access = anything that works is correct

Policy Basic Elements

- ***subject - entity that can "do stuff"***
  - a user or a process representing user/application
- ***object - access controlled resource***
  - e.g. files, directories, records, programs etc
  - NB a subject can also be an object
  - number/type depend on environment
- ***access right - way in which subject accesses an object***
  - e.g. read, write, execute, delete, create, search

What actually is a Policy?

- ***A policy includes at least three components***
  - Targets, Rules, Evaluation procedures, [optionally Obligations],
- ***A Target***
  - (Subject,Action,Object).
- ***A Rule***
  - if Condition is satisfied then applies the Effect (eg permit/deny) upon the Target (i.e. subject executes action on object)".
- ***Evaluation Results***
  - Permit; Deny; Indeterminate; NotApplicableù
- ***Evaluation Procedures (for rule selection)***
  - Deny-overrides; Permit-overrides; First-applicable; Only-one-applicable
- ***Obligations***
  - Security Actions that must be performed after the decision
  - That's bad (we'll see it after)
- ***Abstraction Leves***
  - All possible Entities from Humans → Programs

Types of Policy Rules

- ***Authorization***
  - IF conditions satisfied THEN then grant/deny access
- ***Authorization with Obligations***
  - IF conditions satisfied THEN then grant/deny access
  - AND check user FULFILL Obligation
  - Obligations must be met fulfilled by user AFTER initial access
- ***Examples***
  - Anyone can download free e-books but he should provide his personal information (by filling out a form).
  - Personal information on phone calls in telecommunication systems should be deleted after 3 months.

Example Obligations

- *Time*
  - "file F must be deleted *within 20 days*,"
- *Cardinality*
  - "play game G *at most twice* before it is paid."
- *Event-defined*
  - "*if the data provider revokes document D*, the document must not be used anymore,"
- *Purpose*
  - " *for personal use only*"
- *Environment*
  - "Allow usage *if the firewall is installed*"

What is a "True" Obligation?

- ***Intuitive Obligation:***
  - "I let you do X now but then you will have to do Y in the future"
- ***Intuitive (History Based) Authorization***
  - "I let/don't let you do X now because you have done Y in the past"
- ***BUT***
  - Humans express normally security rules as obligations when they are not really obligations but just wrongly formulated authorizations
- ***That's bad (and makes implementation harder)***
  - We shall see why later in the lecture after we have identified the key components of a IAM
  - In the meanwhile let's try to recognize them first

Test yourself…

- ***What are "True" Obligations?***
  - In digital library system, in order to exercise usage rights users will have to read (click) a non-disclosure agreement.
  - Anyone can download free e-books but he has to provide his personal information (by filling out a form).
  - Users may have to provide usage log information after exercising usage rights
  - Full Professor must present a report of their publications every three years.
  - Borrowed books must be returned in 6 weeks.

Test yourself again

- ***What are "True" Obligations?***

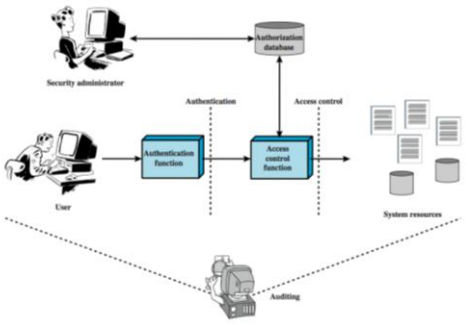
f) Certain information can be read during office hour and usage log has to be reported. In military, officers are allowed to read certain documents only on-site, but if it's not office hour, they have to provide usage log information or fill out a access approval code.

g) In digital library, anyone can download free e-books, but if it's not on-site they have to pay $2 per download.

Key Components of a IAM

- ***Authentication***
  - The verification an identity claimed by (or on behalf of) a system entity
- ***Authorization***
  - The granting of a right (or permission) to the claimant system entity to access a system resource
- ***Audit***
  - The monitoring and processing of user accesses to system resources

Access Control Principles



Identification & Authentication vs Authorization

- ***Identification***
  - Provide the (unique) identifier
    - Example: Social Security Number in Italy (note in US that's messier)
- ***Authentication***
  - Proves that the entity claiming the identifier is actually that entity
    - what you know, what you have, whom you look like, where you are
    - Example: Identity card in Italy, boarding pass in an airport
  - We'll see this in a later lecture
- ***Authorization***
  - Assumes that I&A is already done
  - Applies to subjects, not to users!
  - Decides whether identified principal can get access to resource

Zooming in on the Authorization Module

- ***Architecture specified by the OASIS standard***
  - Done for XML messages
  - Applicable to arbitrary context
- ***Key "Logical" Components***
  - Policy Enforcement Point
  - Policy Decision Point
  - Policy Information Point
  - Policy Administration Point

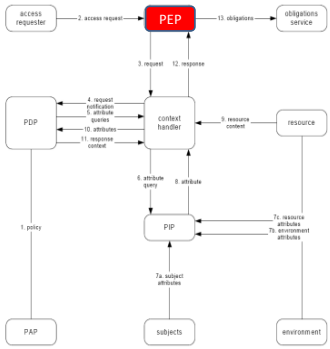
XACML Model's Actors

- ***PAP – Policy Administration Point***
  - The (logical) system entity that creates a ***policy*** or ***policy set***
- ***PEP – Policy Enforcement Point***
  - The (logical) system entity that performs access control, by asking decision requests and enforcing authorization decisions
- ***PDP – Policy Decision Point***
  - The (logical) system entity that evaluates applicable policy and renders an authorization decision
- ***PIP – Policy Information Point***
  - The (logical) entity that acts as a source of attribute values
  - Attributes describing subjects (users), resources, environments (contexts) used to decide whether a control process apply

XACML Main Actors

Policy Enforcement Point

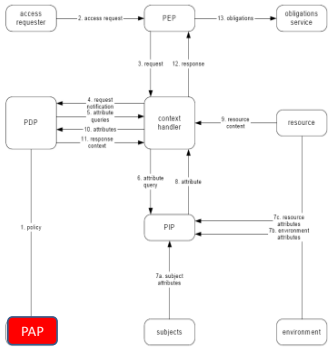
- ***Entity protecting the resource(e.g. file system)***
- ***Performs access control by making decision requests and enforcing authorization decisions and executing obligations***



XACML Main Actors

Policy Administration Point

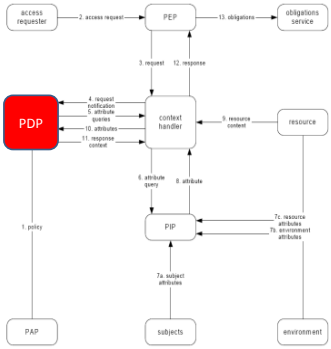
- ***creates security policies and stores these policies in the repository***



XACML Main Actors

The Policy Decision Point

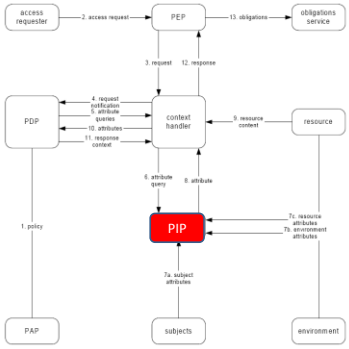
- ***Receives and examines the request***
- ***Retrieves applicable policies***
- ***evaluates the applicable policy and***
- ***Returns the authorization decision to PEP***



XACML Main Actors

Policy Information Point

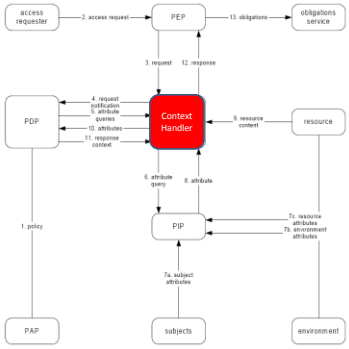
- ***serves as the source of attribute values, or the data required for policy evaluation***



XACML Main Actors

Context Handler

- ***It is the only XML specific actor***
- ***Convert requests in native format → XACML canonical form***
- ***Convert authorization decisions XACML canonical form → native format***
- ***Conceptually irrelevant***



Requirements depend on Scenario

- ***Policy Administration Point***
  - Many distinct entities may act as PAPs – enterprise IT policy, department policy, application-level policy
  - Each entity independently manages its own policies but policies may be linked or depend upon other policies
- ***Policy Enforcement Point***
  - There may be 100s or 1000s or 100.000 or more PEPs in an enterprise
    - E.g. Automated Teller Machines
  - Embedded in devices or applications or infrastructure
  - Some PEPs may need to function in "disconnected mode"

Scenarios Requirements II

- ***Policy Decision Point***
  - For performance and connectivity reasons, there may be multiple PDP instances
  - Performance constraints - some applications require may require 100+ authorization decisions per second with low latency, others only a few decisions per day
- ***Additional Trust and Dependency Issues***
  - How does the context handler obtain needed additional attributes for Resources, Subject, Environment?
  - How to distinguish between attributes originating from the PEP vs. additional attributes needed for policy evaluation?
  - Under what conditions does the PDP and PEP participate in a multi-step interaction?
    - Possibly never (interactions cost time)

Why bother Auths vs Obligs?

- ***Authorization***
  - Use a reactive PEP+ stateless PDP.
  - Easy to implement: If the users don't ask anything you don't need to remember and do anything.
- ***(History Based) Authorization***
  - Use a reactive PEP + stateful PDP.
  - Reasonable to implement: If the users don't do anything you might need to remember something but don't need to do anything
- ***True Obligation:***
  - Use a proactive PEP (obligation monitor) + stateful PDP
  - Costly to implement: even if the user don't do anything you must remember something, monitor users and eventually do something

Results of the test

2009	2010	2011	2014	2015
1-0	0-16	3-6	3-9	0-13
1-2	1-16	2-3	0-14	12-15
5-1	13-3	8-0	12-1	22-0
0-1	14-2	13-2	19-1	29-1
3-0	15-1	16-0	18-0	39-1

What are "True" Obligations?

a) In digital library system, in order to exercise usage rights users will have to read (click) a non-disclosure agreement.
b) Anyone can download free e-books but he has to provide his personal information (by filling out a form).
c) Users may have to provide usage log information after exercising usage rights
d) Full Professor must present a report of their publications every three years.
e) Borrowed books must be returned in 6 weeks.

Results of the test

2009	2010	2011	2014	2015
4-1	12-4	9-2	13-0	22-0
2-1	4-12	1-8	1-8	6.5-15
1-4	11-5	0-9	1-5	7-16

What are "True" Obligations?

f) Certain information can be read during office hour and usage log has to be reported.
g) In military, officers are allowed to read certain documents only on-site, but if it's not office hour, they have to provide usage log information or fill out an access approval code.
h) In digital library, anyone can download free e-books, but if it's not on-site they have to pay $2 per download.

From "false" Obligation to HB Authorization

- ***A False Obligation can always be turned into a History Based Authorization***
  - by shifting the obligation condition on future as a condition on the past
- ***"False" Obligations***
  - in order to exercise usage rights users will have to read (click) license or non-disclosure agreement.
  - Anyone can download free e-books but he should provide his personal information (by filling out a form).
- ***History-based Authorization***
  - users must have previously read (click) license agreement or non-disclosure agreements.
  - To download free e-books you must have previously provided his personal information (by filling out a form).
- ***When you can → always use HB Authorization!***

Some "true" obligations

- ***Examples***
  - Borrowed books must be returned in 6 weeks
    - What if the user doesn't return the book?
  - Certain information can be read during office hour and usage log has to be reported.
    - What if usage log of some actions is not reported?
  - Full Professor must present a report of their publications every three years.
    - What if the professor has no publications?
- ***Alternative Solutions***
  - Implement an obligation monitor or
  - Change the rule (eg by adding the fulfillment of the obligation as a blocking condition for a desired action of the user)

Enforcing Obligations

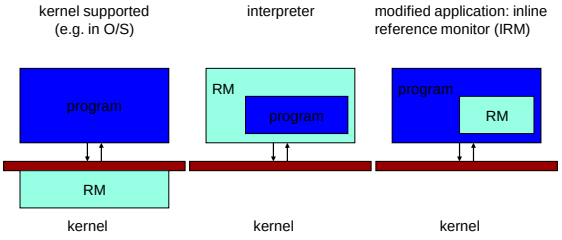
- ***Obligations can/must be enforced at consumer side***
- ***Classification of enforcement mechanisms***
  - Applicability
    - What usages are controlled?
    - What obligations and conditions are supported?
  - Implementation
    - Class of enforcement: inhibition of events, modification (e.g. downgrading the quality), execution of actions (e.g notifying the data owner)
    - Distributed or local?
    - Embedding level: hardware or software?
- ***Example IT Systems***
  - Adobe LiveCycle Management, Windows Media DRM, Amazon E-Book
  - Tivoli Policy Management for Privacy
- ***In your ATM Catalogues***
  - Authorization can be Enforced with Pre-controls
  - Obligation can only be enforced with Post-Controls

Where/How to Actually Deploy PEP?

- ***Critical Step in the control process***
  - It must be NOT-BYPASSABLE
- ***Human Procedure***
  - Airport Security control before going into the gate
    - Generic authorization "Anybody without forbidden items"
    - List of "Forbidden Items" is provided by PAP
    - X-ray scanner provide attributes
    - Security officer at entrance is PDP and PEP
  - Main challenge is social engineering
    - Guard has "preconceptions" → training/randomness to overcome them
- ***IT Procedure?***
  - Many layers to choose from
    - Each abstraction level have different semantics so you can't tell an bowie knife from a cake cutter by looking at individual atoms
  - Can change program API to taylor security need
    - Can't do that with humans: can't change hand so that can only wear a knife to butter the bread. Sometimes true also for IT API

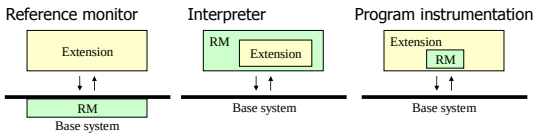
Enforcement Design Choices

- ***Possible choices for a PEP in Case of OS***



Enforcement Design Choices (II)

- ***Reference monitor***
  - may not capture all "high-level" events
  - More difficult to escape
- ***Wrapper/interpreter***
  - performance overhead
- ***Instrumentation: merge monitor into program***
  - different security policies != different merged-in code
  - pay only for what you use
  - Impossible for humans
- ***What happens if things don't work? Is the program or the security fault?***



Enforcement Design Choices (III)

- ***Reference Monitor as the "Default" PEP***
  - Observes the execution of a program/process and halts the program if it's going to violate the security policy.
- ***Most enforcement mechanisms are reference monitors***
  - They are "simple" to build and understand
  - But can miss the semantics of events
- ***Common Examples:***
  - O.S. memory protection
  - Access control checks
  - Routers and Firewalls
  - Security officer at airport gates

Types of Access Control

- ***Discretionary Access Control***
  - Policy decided by individual subjects
  - Access based on identity of subjects
- ***Role based Access Control***
  - Policy decided by system
  - Subjects assigned to Roles, (Action,Objects) assigned to Roles
  - Access based on roles activated by subjects
- ***Mandatory Access Control***
  - Policy decided by system
  - Subject assigned to security levels (clearance), Object assigned to security labels
  - Access based on matching objects' labels to subjects' clearances
- ***Credential based Access Control***
  - Access based on attributes qualifying a subject
    - Essentially the subject sends the policy applicable to him attached to the request itself ("self-service" PIP) and signed by accredited PAPs

Discretionary Access Control

- ***Intuitions:***
  - Owners of resources decide who can access them
- ***Intended Environment:***
  - Operating systems in the late 60s (Lampson, 71)
  - Users are members of the same community: objective is to protect data from mistakes of others (cd /; rm -fr *)
- ***Entities***
  - Subjects: who detain privileges and can do actions
  - Objects: files, resources, programs
  - Actions: what subject can do to objects…
- ***Authorization State***
  - Specify on (Subject x Object) basis what can be done

Discretionary Access Control

- ***Often provided using an access matrix***
  - lists subjects in one dimension (rows)
  - lists objects in the other dimension (columns)
  - each entry specifies access rights of the specified subject to that object
- ***Access matrix is often sparse***
  - can decompose by either row (Capabilities) or column (Access Control Lists)
- ***Even in this simple model security of administrative changes is undecidable***

An example of DAC Model

		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

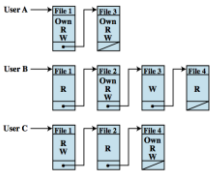
Figure 4.4 Extended Access Control Matrix

Access Control Structures

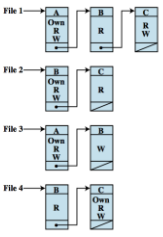
ACCESS MATRIX

	OBJECTS File 1	File 2	File 3	File 4
SUBJECTS User A	Own Read Write		Own Read Write	
User B	Read	Own Read Write	Write	Read
User C	Read Write	Read		Own Read Write

CAPABILITY LIST



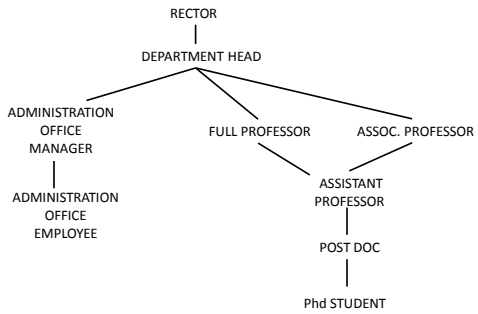
ACCESS CONTROL LIST



Role based access control (RBAC)

- ***Widely adopted access control model mostly for ERP systems***
- ***Based on the role played by a user within an organization***
- ***Ideas***
  - Roles are assigned access rights to resources
  - Users are assigned to roles
    - Inherit access rights of the role they play
  - Possibly add constraints or inheritance

RBAC+ Role Hierarchy



RBAC++ Constraints

- ***Mutually exclusive roles***
  - One user can be assigned to only one role
  - A permission can be granted to only one role
- ***Cardinality***
  - Maximum number of users that can activate a role
  - Maximum number of roles that can be played by a user
  - Maximum number of roles that can be granted a permission
- ***Prerequisites***
  - A user is assigned to a role only if it played another role

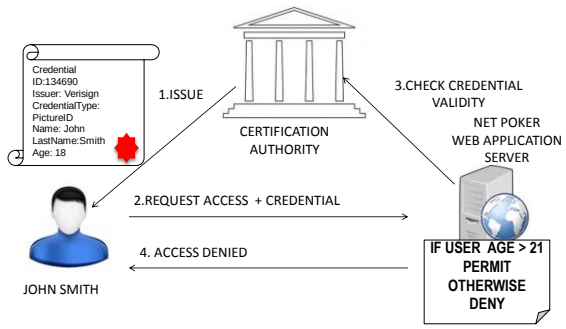
Static & Dynamic Separation of Duty

- ***Static Separation of Duty***
  - pair (role set, n)
  - a user cannot be assigned to more than n roles in the role set
- ***Dynamic Separation of Duty***
  - pair (role set, n)
  - a user cannot activate more than n roles in the role set within the same session

Credential-based Access Control

- ***Credential***
  - assertion by a certification authority that a subject hold certain attributes
  - e.g Bob's credit card number = 418789 , card type = visa
  - e.g Alice's national identification number = FP291178D78F20
- ***Access control policies specify conditions against subject attributes***
  - All the users older than 21 can access from NetPoker website

Credential-based Access Control



An Exercise in Thinking

- ***Auguste Kerckhoffs, « La cryptographie militaire »,***
  - Journal des sciences militaires, vol. IX, pp. 5–38, Janvier 1883, pp. 161–191, Février 1883.
- ***Five Principles Mostly Used by Cryptographers***
  - The system must be substantially, if not mathematically, undecipherable;
  - The system must not require secrecy and can be stolen by the enemy without causing trouble;
  - It must be easy to communicate and remember the keys without requiring written notes, it must also be easy to change or modify the keys with different participants;
  - The system ought to be compatible with telegraph communication;
  - The system must be portable, and its use must not require more than one person;
  - Finally, regarding the circumstances in which such system is applied, it must be easy to use and must neither require stress of mind nor the knowledge of a long series of rules
- ***How to Adapt them to Access Control?***

A key suggestion

- ***Auguste Kerckhoffs, « La cryptographie militaire », Journal des sciences militaires, vol. IX, pp. 161–191, Février 1883.***
  - Il dépendra donc de l'Administration d'assurer l'avenir de la cryptographie militaire, en n'accordant ses suffrages qu'à l'invention qui s'appuiera sur le principe que du Carlet, un des maîtres de notre art au xviie siècle, avait inscrit comme devise en tête de sa méthode [26], principe qui résume d'ailleurs toute ma thèse, à savoir qu'un chiffre n'est bon qu'autant qu'il reste indéchiffrable pour le maître lui-même qui l'a inventé : Ars ipsi secreta magistro.
- ***A cipher is good only if it remains undeciphrable for the very designer who invented it***
- ***A security solution is good only it it is secure also against those who invented it***

Reading Material

- ***XACML v3 Core Specification. Available at:***
  - http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.pdf
- ***Chapter 4. W.Stallings and L. Brown. Computer Security. Principles and Practices***
- ***Various reading papers on access control failures***