Policy Machine

Towards a unifying access control mechanism

David Ferraiolo
National Institute of Standards and Technology

dferraiolo@nist.gov

Access Control: State of Practice

- The ability to control access to sensitive data in accordance with policy is perhaps the most fundamental security requirement
- However, specification and enforcement of enterprise policy remains in a dismal state of affairs.
- Most approaches have been ad hoc or have focused on management issues and/or a specific policy problems and/or environments
- Controls as implemented are not comprehensive, typically do not offer control at the process/inter-process level, and/or lack expressive power.
- For instance, a user with read access to data can typically make a copy of that data and paste its contents into an email message and send it to anyone else in the world, regardless of enterprise policy, and a user process can can do anything its user can without the user’s knowledge.

Policy Problem

- While over the past four decades a large variety of policies and policy models have been proposed to address real world security issues, only a small subset of these policies are enforceable through off-the-shelf technologies
- Writing down policy and faithfully enforcing policy are different things!

Interoperability Problem

- A natural characteristic of dispersed heterogeneous mechanisms is a lack of interoperability.
- This lack of interoperability results in many of the identity and policy (privilege) management issues that enterprises struggle with today, as well as the lack of comprehensive policy enforcement.
- Email and external storage devices are big holes
- Processes and inter-process communication (e.g., copy and paste) are most often not controlled

Access Control Mechanisms are Logical Machines

- OS or application access control mechanisms can be thought of and analyzed as a complete and logically independent machine to that of its host environment
- Each of these machines include:
  - Access control data for the expression of policy, and
  - A set of functions for
    - computing access control decisions based on the configuration of the data and
    - Enforcement of policy based on the decisions
- Problem:
  - Each mechanism expresses policy, computes decisions and enforces policy differently

Interoperability is solvable through a standardized architectures

- A family of standards that recognizes policy enforcement points (PEP), Policy Decision Point(s) (PDP), a Policy Administration Point (PAP), and a policy database
- Each component has standardized functions and APIs and the data has been standardized
- Not really new, e.g., XACML

But, what about policy

- Before solving the inter-operability problem, we must first solve the policy problem
- Policy enforcement drives access control data and functions, and data and functions drive standards
- To proceed otherwise would be arbitrary
- Prominent privilege management technologies and standards include DAC/ACLs, RBAC, ABAC/XACML, MAC, DTE
- Each has advantages and disadvantages

The need for a Unifying Policy Machine

A logical “machine” comprising of a fixed set of data relations, configurable through a fixed set of administrative operations for the expression of **combinations of any policy**, and a fixed set of functions for making access control decisions, and enforcing policy based on that expression.

Why is a Policy Machine important?

- One generic mechanism for comprehensive enforcement of many policies, providing
- A single administrative domain and scope of control that extends over a multitude of OSs, devices, applications, and data that can be scattered over a multitude of organizational entities forming a virtual enterprise for secure access and sharing data

Can a Policy Machine be developed?

- We think so, at least for attribute-based policies.
- We have identified a surprisingly small set of data relations and functions that are re-usable in the expression and enforcement of a wide range of policies
- PM data configurations specify capabilities that users and processes **“can”** perform (under assignment relations), and **“can not”**, and **“can only”** perform (under prohibition relations). In addition to these relations, the PM defines obligations that dynamically change the policy state in response to user/process events, and in doing so, specifies capabilities that users and/or processes **“can now”**, **“can no longer”**, or **“now can only”** perform.
- These assignment, prohibition and obligation relations have been shown to provide the basic ingredients for expressing a wide range of attribute-based policies.

Our Reference Implementation

- We can demonstrate the expression and enforcement of a wide variety of policies (e.g., instances, combinations and hybrids of DAC, MAC, RBAC, Chinese wall, ORCON, history-based Separation of duty, OMB M-07-16, etc.)
- Policies are not only enforced on files, but comprehensively enforced across a rich user environment that includes the Open Office suite of applications, email, workflow, records, and forms management, and Copy/Cut & Paste

Questions?