Security Policy Interoperability

Working with Policies Across Organizations

27 February 2001
### Agenda

- What is a Security Policy Language
- What will it do for me?
- Some limitations of Security Policy Languages
- Statement of the Problem
- What is needed
- The Coalition Security Policy Language(s) Project
- Technical Approach
- Participation
What is a Security Policy Language?

- **Policy** – A rule that can be used to change the behavior of a system, domain, organization, or component.
- **Policy Language** – A framework for specifying policies (rules) that is independent of entity or entities implementing those policies.
- **Security Policy Language** – A framework for specifying the rules including, but not limited to:
  - Identification & Authorization – what entities external to the subject security domain can have access to information/services offered within the security domain and what limitations are placed upon that access
  - Obligation – the actions of entities within the security domain with respect to other entities, both within and external to the security domain
  - Restraints – the actions that an entity may not take (negative authorizations, in a way)
  - Delegations – what actions can be delegated to other entities
  - Constraints – limits to the applicability of policies that based on variable parameters
What will it do for me?

- Allow specification and modification of policies independent from implementation
  - Dynamic modifications
  - No recoding
- Persistence of policies
- Precise specification of subjects, targets, actions, and constraints
  - Applicable to humans and automated entities
  - Defines responsibilities, rights, and duties based on roles
    - Roles of individuals
    - Roles of groups
    - Roles of individuals relative to groups (role relationships)
    - Role specialization
    - Role generalization
- Allows adjustment of Quality of Service (QoS) as roles and context change
- Reuse of policies and sets of policies
  - Policy patterns?
Some Limitations of Security Policy Languages

- Single regulatory, linguistic, and cultural context
  - Laws and regulation differ from country to country
  - Language and cultural differences cause semantic disconnects

- Conflict detection and resolution
  - Multiple Policies
    - An entity may be a member of multiple domains with conflicting policies
    - Multiple roles with conflicting policies
  - Policy Precedence
    - Group vs. role vs. entity
  - Semantic Conflicts
    - Separation of duties
    - Self-management
    - Resource conflict

- Not interoperable
  - No single accepted ontology

- Domains must have prior knowledge of each other
Statement of the Problem

- Successful inter-enterprise operations depend on effective information exchange
  - Security and privacy practices and procedures vary nation to nation and enterprise to enterprise
    - Security laws & regulations differ in each nation
    - Privacy laws & regulations differ in each nation
    - Security and privacy ethics vary culturally
      - Not necessarily nation to nation
        - Canada and Belgium are examples
        - Each enterprise has its own culture
  - Resolution of differences can take up to 6 months to negotiate

- Automation makes it more difficult
  - Software does not make “judgement calls”
  - No mechanism for exchanging security policy information between domains unless:
    - A security policy language is used
    - The same security policy language with a well defined ontology is used
    - The same semantics and heuristics apply

Key concept: It must be possible to exchange some level of information between entities that have no prior knowledge or experience with each other.
- Simple case requires that the parties have pre-established trust relationship
- N-party introductions require that the members of each 2 party association have a pre-established trust relationship
- No pre-established trust relationship $\rightarrow$ NO exchange
What is Needed

- Do you want to exchange information?
  - Yes? Do I know you?
    - Yes? Okay we can communicate.
  - No? Is there someone that can do a proper introduction?
    - Yes? Okay we can communicate as soon as we are introduced.
  - No? Maybe there are some things that we can talk about.
    - Tell me a little about yourself
      - Do we have common friends?
      - Do we have similar policies?
      - What is your role?

Key concept: Security Policy Languages should always be role based.
The Coalition Security Policy Language(s) Project

- Identify a means of exchanging security policy information
  - Universally understood form
    - UML
    - XML
    - Other?
  - Commonly understood ontology and agreed upon heuristics
    - Exchange knowledge of security policies
    - For negotiating parameters of information exchange

- Primary Goals
  - Year 1: Enable local server administrator to grant accesses to previously unknown clients based on verifiable policies
  - Year 2: Automate negotiation process to grant access to previously unknown clients based on verifiable policies
Technical Approach – Year 1

• MITRE led, but augmented by others in US and Non-US
  – Examine a representative set of policy languages including, as a minimum:
    • Ponder -- Imperial College, University of London)
    • Domain and Type Enforcement Language (DTEL++) -- Network Associates, Inc.
    • Language for Security Constraints on Objects (LaSCO) -- University of California, Davis)
    • Security Policy Specification Language (SPSL) draft-ietf-ipspspsl-00 -- Verizon/BBN
    • Policy Language for AuthorizationS (PLAS), IBM Zurich Research Laboratory
    • Trust Policy Language (TPL), IBM
    • Security Policy Language (SPL), Mageland Institute, Portugal
    • Policy Description Language (PDL)
  – Examine other relevant research, including:
    • Defense Advanced Research Projects Agency (DARPA)
    • Computer Science Department, Friedrich-Alexander University
    • Institut fur Informatik Frie, University of Berlin
    • Distributed System Technology Center (DSTC), University of Queensland
    • Center for IT Integration, University of Michigan
  – Define security policy information interchange mechanism
    • Identify or develop object model, DTD, and/or other construct
    • Identify and document API(s) for generating and parsing
      – Formal specification appropriate for submission to OMG, W3C, IETF, ISO, or some other appropriate technology/standards adoption organization
  – Prototyping
    • Software for interchange
    • Software to allow security or system administrator to determine access rights of requester
Technical Approach – Year 2

• Submit specification to appropriate technology/standards adoption organization(s) for adoption
  – Fast track probable

• Refine exchange mechanism
  – Document in new version of specification
    • Submit to appropriate technology/standards adoption organization(s) for adoption

• Evaluate Knowledge Management (KM) tools and techniques
  – Identify tool or technique to use to automate negotiation of access rights
    • Use APIs defined in Year 1
    • Define and document additional APIs as needed in new specification(s)
    • Submit new specifications to appropriate technology/standards adoption organization(s) for adoption

• Prototype negotiation functionality
• Build pilot implementation(s)
Participation - US

- **Specification development leads**
  - MITRE Corporation (Project Lead)
    - 4 Staff at ½-time each
      - Shel Sutton (PI)
      - Chuck Heazel (Associate PI)
      - Dock Allen (real-time security)
      - Don Faatz (Security Center)
  - Center for Systems Assurance, Syracuse University (TBR) (Polar Humenn, PI)

- **Prototype/pilot development leads**
  - Adiron (Polar Humenn, PI)
  - 2AB is considering participation
  - Others are considering participation

- **Integration, verification, and validation leads**
  - Cherokee Information Services (Pug Gutridge, PI)
  - At least one large integrator desired
    - EDS is considering participation

*Note: The fact that a company is considering participation should not be taken to imply any commitment. Further discussions are pending.*
Participation – Non-US

• Specification development leads
  – Imperial College, University of London (TBR) (UK -- Prof. Morris Sloman, PI)
  – 1 or 2 additional organizations desired
    • DERA (UK) is considering participation
    • Open-IT (UK) is considering participation
    • DSTO (Australia) is considering participation
    • SINTEF (Norway) is considering participation

• Prototype/pilot development leads
  – OSM SARL (France -- Stephen McConnell, PI)
  – 1 to 3 additional organizations desired
    • Open-IT (UK) is considering participation
    • DSTO (Australia) is considering participation
    • SINTEF (Norway) is considering participation

• Integration, verification, and validation leads
  – 2 or 3 organizations desired
    • HM&V Research (Finland) is considering participation

Note: The fact that a company is considering participation should not be taken to imply any commitment. Further discussions are pending.
Interested in participating? Have questions? Etc.?

• Shel Sutton
MITRE Corporation
1820 Dolley Madison Blvd.
McLean, VA 22102

• Telephones
  – Office: +1 703-883-6677
  – Mobile: +1 703-869-6677
  – Home: +1 703-281-2852

• Faxes:
  – Office: +1 703-883-6801
  – Home: +1 703-938-4335

• Email:
  – Office: shel@mitre.org
  – Other: shels@flash.net
cspl@shel.com (after 3/4/01)

• Web
  – http://www.shel.com/cspl.htm (currently under construction)
Backup Slides
Abstract:
Many emerging applications on the Internet requiring group communication have varying security requirements. Significant strides have been made in achieving strong semantics and security guarantees within group environments. However, in existing solutions, the scope of available security policies is often limited. This paper presents the Antigone framework. Antigone provides a suite of mechanisms from which flexible application security policies may be implemented. Using this approach, developers may chose a policy that best addresses their security and performance requirements. We describe the Antigone’s mechanisms, consisting of a set of micro-protocols, and show how different security policies can be implemented using those mechanisms. We also present a performance study illustrating the security/performance tradeoffs that can be made using Antigone.
Abstract:

The Language for Security Constraints on Objects (LaSCO) is a new approach to expressing security policies using policy graphs and present a method for enforcing policies so expressed. A security policy is a statement about how a system (any executing entity) should behave with respect to a site’s particular notion of security. Other approaches for stating security policies fall short of what is desirable with respect to either policy clarity, executability, or the precision with which a policy may be expressed. This results in expressed policies that are ambiguous, are not implementable, or are that are not an accurate reflection of the policy goal, respectively. However, LaSCO is designed to have those three desirable properties of a security policy language as well as: relevance for many different systems, statement of policies at an appropriate level of detail, user friendliness for both casual and expert users, and amenability to formal reasoning. In LaSCO, the constraints of a policy are stated as directed graphs annotated with expressions describing the situation under which the policy applies and what the requirement is. LaSCO may be used for such diverse applications as executing programs, file systems, operating systems, distributed systems, and networks.

Formal operational semantics have been defined for LaSCO. An architecture for implementing LaSCO on any system, consisting of a system-independent policy interpretation engine and a system-specific interface layer, is presented.
Abstract:

The Ponder language provides a common means of specifying security policies that map onto various access control implementation mechanisms for firewalls, operating systems, databases and Java. It supports obligation policies that are event triggered condition-action rules for policy based management of networks and distributed systems. Ponder can also be used for security management activities such as registration of users or logging and auditing events for dealing with access to critical resources or security violations. Key concepts of the language include roles to group policies relating to a position in an organisation, relationships to define interactions between roles and management structures to define a configuration of roles and relationships pertaining to an organisational unit such as a department. These reusable composite policy specifications cater for the complexity of large enterprise information systems. Ponder is declarative, strongly-typed and object-oriented which makes the language flexible, extensible and adaptable to a wide range of management requirements.