Semantic Web Policy Systems

Presented By: John Paul Dunning

Semantic Web Policy Systems

"A meta-control architecture for orchestrating policy enforcement across heterogeneous information sources"

Jinghai Rao, Alberto Sardinha, Norman Sadeh

Carnegie Mellon University

ELSEVIER

Available online at www.sciencedirect.com
ScienceDirect
Web Semantics: Science, Services and Agents on the World Wide Web 7 (2009) 40–56

JOURNAL OF Web Semantics
www.elsevier.com/locate/websem

A meta-control architecture for orchestrating policy enforcement across heterogeneous information sources

Jinghai Rao, Alberto Sardinha, Norman Sadeh *

School of Computer Science, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, USA

Received 18 August 2007; accepted 25 October 2007
Available online 3 December 2007

Abstract

There is increasing demand from both organizations and individuals for technology capable of enforcing sophisticated, context-sensitive policies, whether security and privacy policies, corporate policies or policies reflecting various regulatory requirements. In open environments, enforcing such policies requires the ability to reason about the policies themselves as well as the ability to dynamically identify and access heterogeneous sources of information. This article introduces a semantic web framework and a meta-control model to orchestrate policy reasoning with the identification and access of relevant sources of information. Specifically, sources of information are modeled as web services with rich semantic profiles. Policy Enforcing Agents rely on meta-control strategies to dynamically interleave semantic web reasoning and service discovery and access. Meta-control rules can be customized to best capture the requirements associated with different domains and different sets of policies. This architecture has been validated in the context of different environments, including a collaborative enterprise domain as well as several mobile and pervasive computing applications deployed on Carnegie Mellon's campus. We show that, in the particular instance of access control policies, the proposed framework can be viewed as an extension of the XACML architecture, in which Policy Enforcing Agents offer a particularly powerful way of implementing XACML's Policy Information Point (PIP) and Context Handler functionality. At the same time, our proposed architecture extends to a much wider range of policies and regulations. Empirical results suggest that the semantic framework introduced in this article scales favorably on problems with up to hundreds of services and tens of service directories.

Keywords: Semantic web; Context-sensitive policies; Web services; Security and privacy

1. Introduction

The increasing reliance of individuals and organizations on the Web to help mediate a variety of activities is giving rise to a demand for richer policies (e.g. security and privacy policies as well as other corporate or regulatory policies) and more flexible mechanisms to enforce these policies. People may want to selectively expose sensitive information to others based on the evolving nature of their relationships, or share information about their activities under particular conditions. This trend requires context-sensitive security and privacy policies, namely policies whose conditions are not tied to static considerations but rather conditions whose satisfaction, given the very same actors (or principals), will likely fluctuate over time. Enforcing such policies in open environments is particularly challenging for several reasons:

- sources of information available to verify these policies may vary from one principal to another (e.g. different users may have different sources of location tracking information made available through different cell phone operators);
- available sources of information for the same principal may vary over time (e.g. when a user is on company premises her location may be obtained from the wireless LAN location tracking functionality operated by her company, but, when she is not, this information can possibly be obtained via her cell phone operator);
- available sources of information may not be known ahead of time (e.g. new location tracking functionality may be installed or the user may roam into a new area).

Accordingly, enforcing context-sensitive policies in open domains requires the ability to opportunistically interleave policy reasoning with the dynamic identification, selection and access of relevant sources of contextual information. This requirement, which is not unique to domains with context-

* Corresponding author.
E-mail addresses: jinghai@cs.cmu.edu (J. Rao), alberto@cs.cmu.edu (A. Sardinha), sadeh@cs.cmu.edu (N. Sadeh).

1570-8268/$ – see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.websem.2007.10.001

Overview

- Context-sensitive security and privacy policies
- Decentralized trust management
- Challenges include:
  - sources of information vary from one principal to another
  - sources of information may vary over time
  - sources of information may not be known ahead of time

Contributions of Paper

- *"Development of a semantic web framework and a meta-control model for opportunistically interleaving policy reasoning and web service discovery to enforce context-sensitive policies"*
- Extension of XACML ontology
- Language independent system

XACML

- “XACML is an initiative to develop a standard for access control and authorization systems...
- XACML aims to achieve the following:
  - Create a portable and standard way of describing access control entities and their attributes.
  - Provide a mechanism that offers much finer granular access control than simply denying or granting access -- that is, a mechanism that can enforce some before and after actions along with "permit" or "deny" permission.”

http://www.ibm.com/developerworks/xml/library/x-xacml/

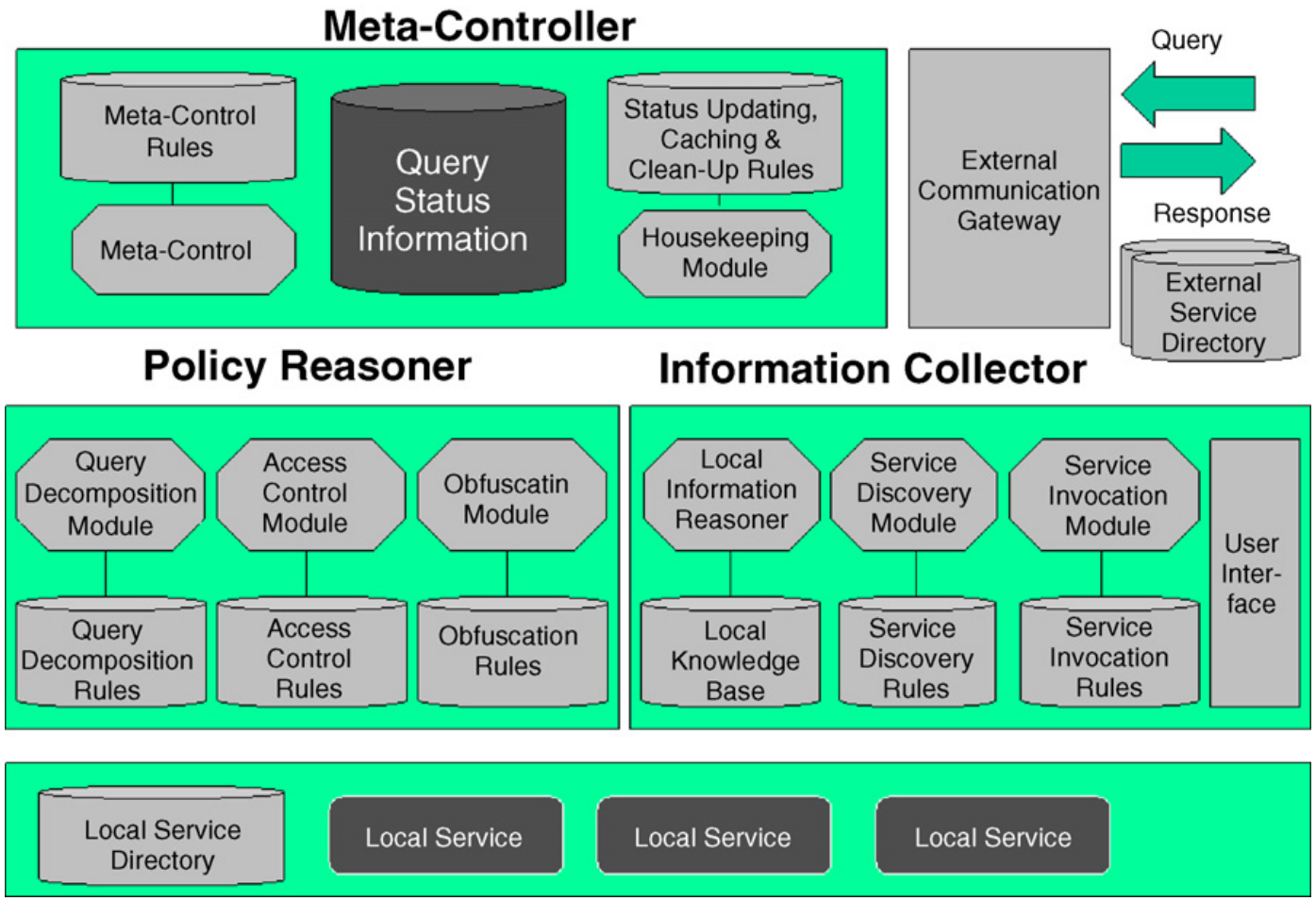
Information Disclosure Agent (IDA)

- Policy Enforcement Agent (PEA)
- Controls access to information and service access through policies
- Uses policy enforcement
  - Control policies
  - Obfuscation policies

Information Disclosure Agent (IDA)

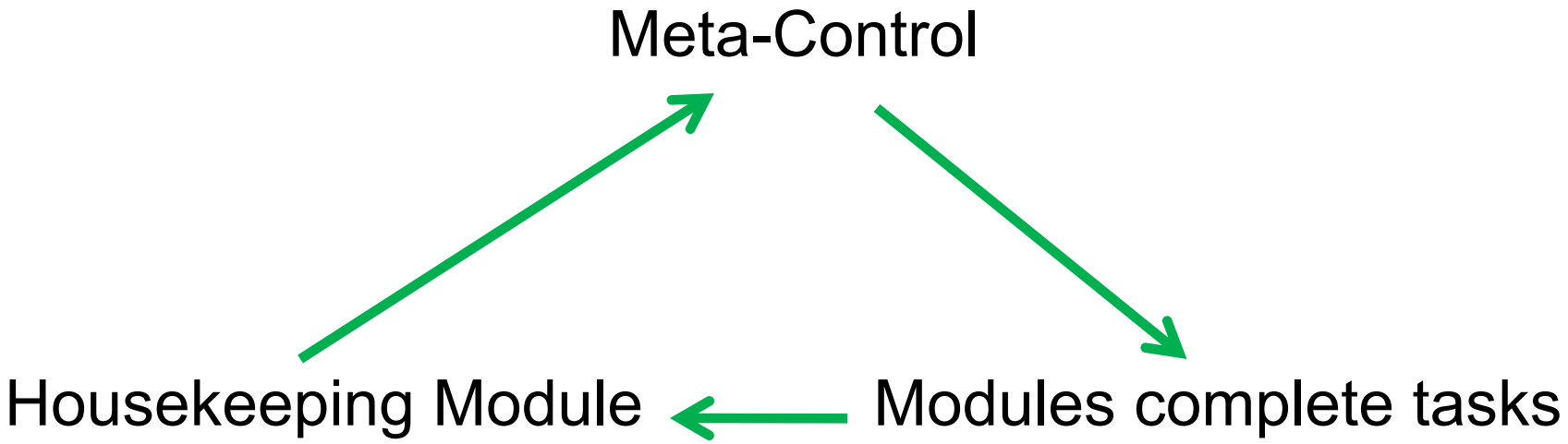
- Interact across various networks
- Encrypted traffic
- Language Independent (with interpreter)

Information Disclosure Agent (IDA)



Meta-Controller

- Monitors progress and determines the next step
- Cycle

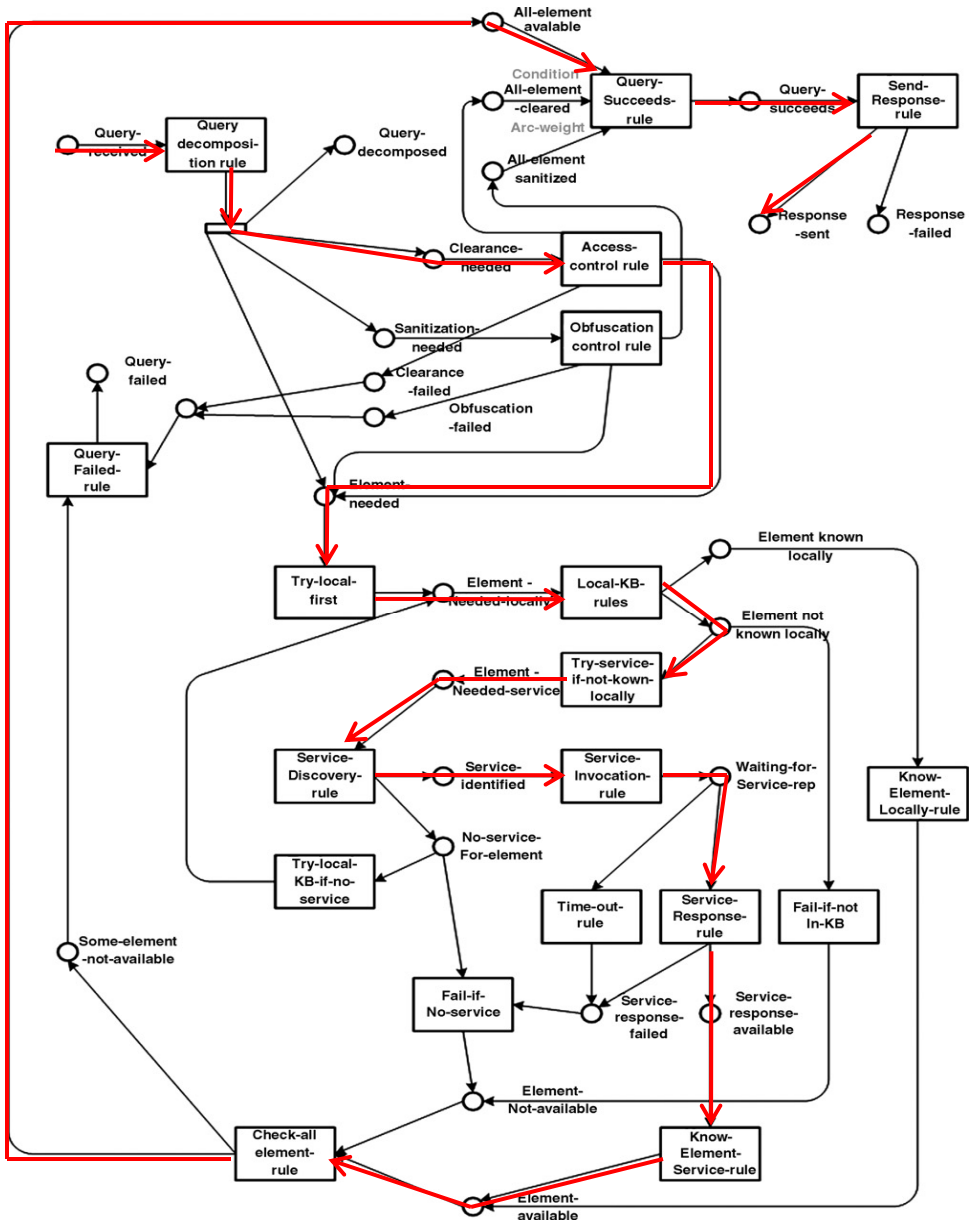


Meta-Controller

- Query status information includes:
  - *A query status ID*
  - *Status predicates*
  - *A query ID and query element ID*
  - *A parent query status ID*
  - *A time stamp*

Meta-Controller

	Sample Status Predicates	Description
(1) communication status predicates	Query-Received	A particular query has been received.
	Sending-Response	Response to a query is being sent
	Response-Failed	Response failed (e.g. message bounced back)
(2) query status predicates	Processing-Query	Query is being processed
	Potentially-deadlocked-query	Used to flag queries that may correspond to possible deadlocks
	Query-Decomposed	Query has been decomposed (into primitive query elements)
	Query-Succeeded	All query elements are available and cleared. Ready to send response.
	Query-Failed	Some query elements are not available or cleared.
	All-Elements-Available	All query elements associated with a given query are available (i.e. all the required information is available)
	All-Elements-Cleared	All query elements have been cleared by relevant access control policies
	All-Elements-Sanitized	All query elements have been sanitized according to relevant obfuscation policies
	Query-make-deadlock	The incoming query may result in an endless loop. According to different meta control rules, the IDA may respond a failure to query sender, or consult the user to handle the problem.


All-element avalable
Condition
All-element -cleared
Query-Succeeds-rule
Query-succeeds
Send-Response-rule
Arc-weight
Query-received
Query decomposi-tion rule
Query-decomposed
All-element sanitized
Response -sent
Response -failed
Clearance-needed
Access-control rule
Sanitization-needed
Obfuscation control rule
Query-failed
Clearance -failed
Obfuscation -failed
Query-Failed-rule
Element needed
Element known locally
Try-local-first
Element -Needed-locally
Local-KB-rules
Element not known locally
Try-service-if-not-kown-locally
Element -Needed-service
Service-Discovery-rule
Service-identified
Service-Invocation-rule
Waiting-for-Service-rep
Know-Element-Locally-rule
No-service-For-element
Try-local-KB-if-no-service
Time-out-rule
Service-Response-rule
Fail-if-not In-KB
Some-element -not-available
Fail-if-No-service
Service-response-failed
Service-response-available
Element-Not-available
Check-all element-rule
Know-Element-Service-rule
Element-available

Policy Reasoner

- Evaluating relevant policies
- Return policy decisions
- Modules:
  - Query Decomposition Module
  - Access Control Module
  - Obfuscation Module

Information Collector

- Gathering facts
- Modules:
  - Local Information Reasoner
  - Service Discovery Module
  - Service Invocation Module
  - User Interface

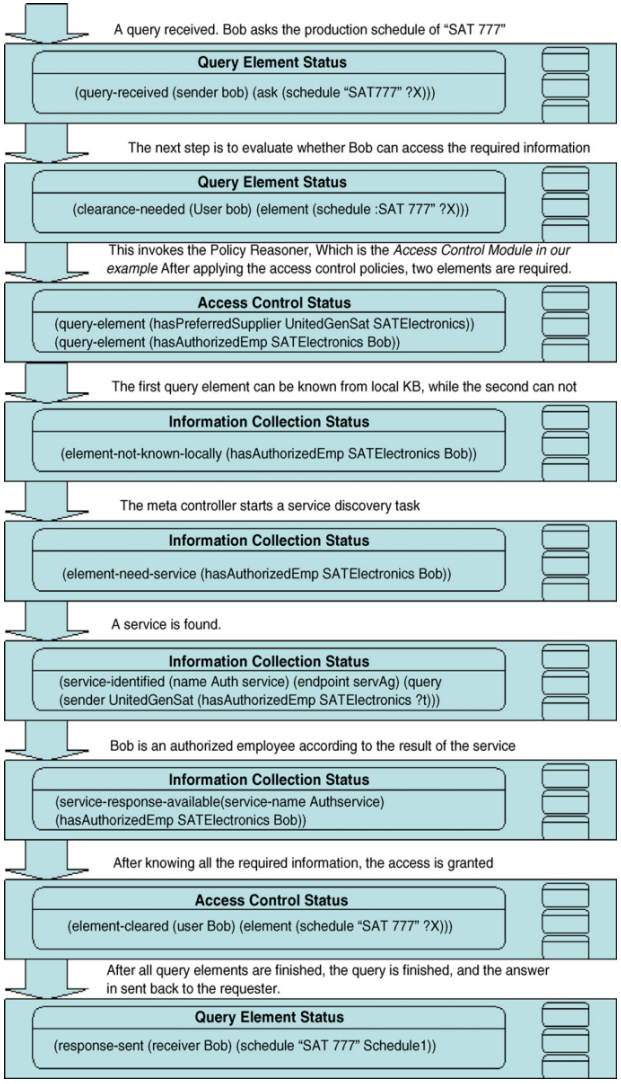
Service Discovery and Invocation

- IDAs are constantly sending queries and results back and forth
- Multiple queries between IDAs
- Node deadlock is possible and avoidable
  - Time outs
  - Query dependency graphs

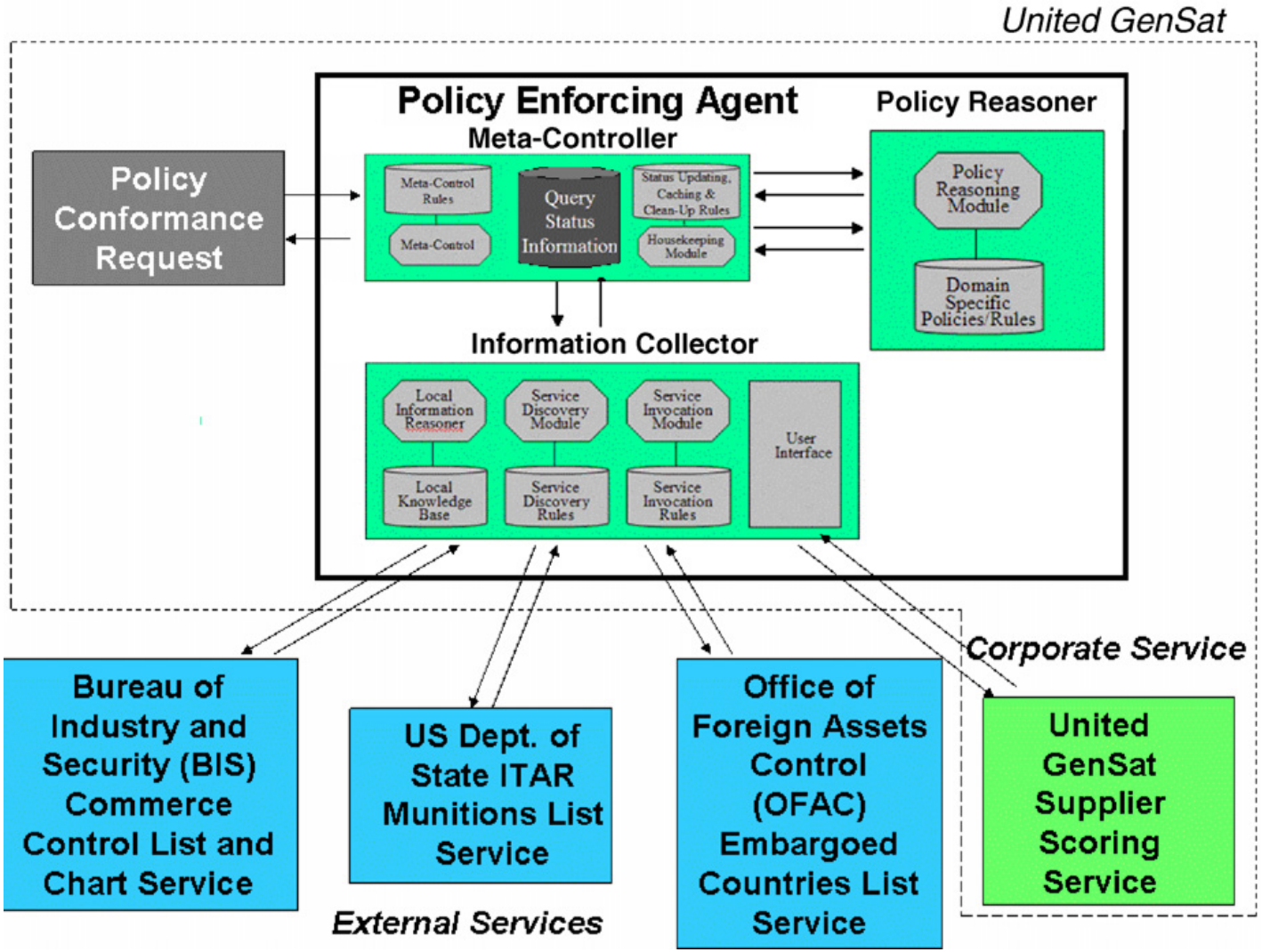
Example Scenario

- Bob is an employee of SATElectronics Corporation
- Bob contracts to United GenSat
- Bob wants the schedule for deployment of SAT 777 from United GenSat, which is a product he has been working on.

Example Scenario



Beyond Access Control Policies



Q&A

- How easy are the policies to create/update/delete?
- What is the overhead of this system VS a standard form of authentication?