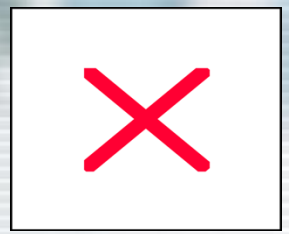


A Policy Based Approach to Security for the Semantic Web

Lalana Kagal, Tim Finin and Anupam Joshi

UMBC
AN HONORS UNIVERSITY IN MARYLAND

ebiquity

Outline

- Rei : A policy language
- Why is Rei needed ?
  - Comparison with existing research
- Securing the Semantic Web
  - Infrastructure for web resources
  - Infrastructure for agents
  - Infrastructure for web services
- Summary

Rei : A Policy Language

Japanese 'Kanji' character means 'universal' or 'essence'
Kanji is a Japanese script

- A declara[illegible]olicy language f[illegible]escribing policies ov[illegible]tions
- Represente[illegible]n RDF-S + logic
- Based on deontic concepts and
- Possible to write Rei policies over ontologies in other semantic web languages
  - Rei policy engine + RDFS reasoner + other reasoners
- Different kinds of policies
  - Security, privacy, conversation, etc.

Right
Prohibition
Obligation
Dispensation

Delegation
Revocation
Request
Cancel

Example : All entities in the same group as John h[illegible]

Example : John has the right to delegate the right to revoke the right to Print

Why is it needed ?

- Existing policy languages
  - XACML : OASIS eXtensible Access Control Markup Language
  - Ponder
  - EPAL : IBM Enterprise Privacy Authorization Language
  - KeyNote
  - KAoS : Knowledgeable Agent-oriented System

- Disadvantages
  - Limited by language used
  - Not very expressive in terms of constraints
  - Limited support for delegation
    - Other speech acts not handled at all

Rei

RDFS

Expressive

Good delegation
mng+ integrated
support for
other speech acts

Rei Specifications

- Policy
  - Properties : Context, Default Policy, Grants
- Deontic objects
  - Rights, Prohibitions, Obligations, Dispensations
  - Properties : Actor, Action, Constraints
- Actions
  - Properties : Actor, Target objects, PreConditions, ...
  - C...
- Spe...
  - Delegation, Revocation, Request, Cancel
  - Properties : Sender, Receiver, Deontic object/Action
  - Used to modify policies

Example : No student can enter the faculty lounge after 4.30 on weekdays

Example : John is prohibited from any action that causes radiation

Right

Prohibition

...bligation/ ...elegation

...ensation/ ...ocation

Rei Specifications

- Meta Policies
  - Setting priorities between policies or rules
    - *E.g. Federal policy overrides the State policy*
  - Setting modality precedence
    - *E.g. Negative modality holds for all students of UMBC*

Security framework

- Provide security for three types of entities
  - Web resources
  - Agents
  - Web services

Classification of entities

- Entities can be one of 3 types
  - Private -- No other entity has the right to access a *private* service/agent/resource
  - Secure -- Only entities that satisfy the associated policy of the *secure* agent/service/resource have the right to access it
  - Open -- All entities have the right to access an *open* resource/service/agent

Framework for web resources

Web server

Request for resource

User

Resource

Agent

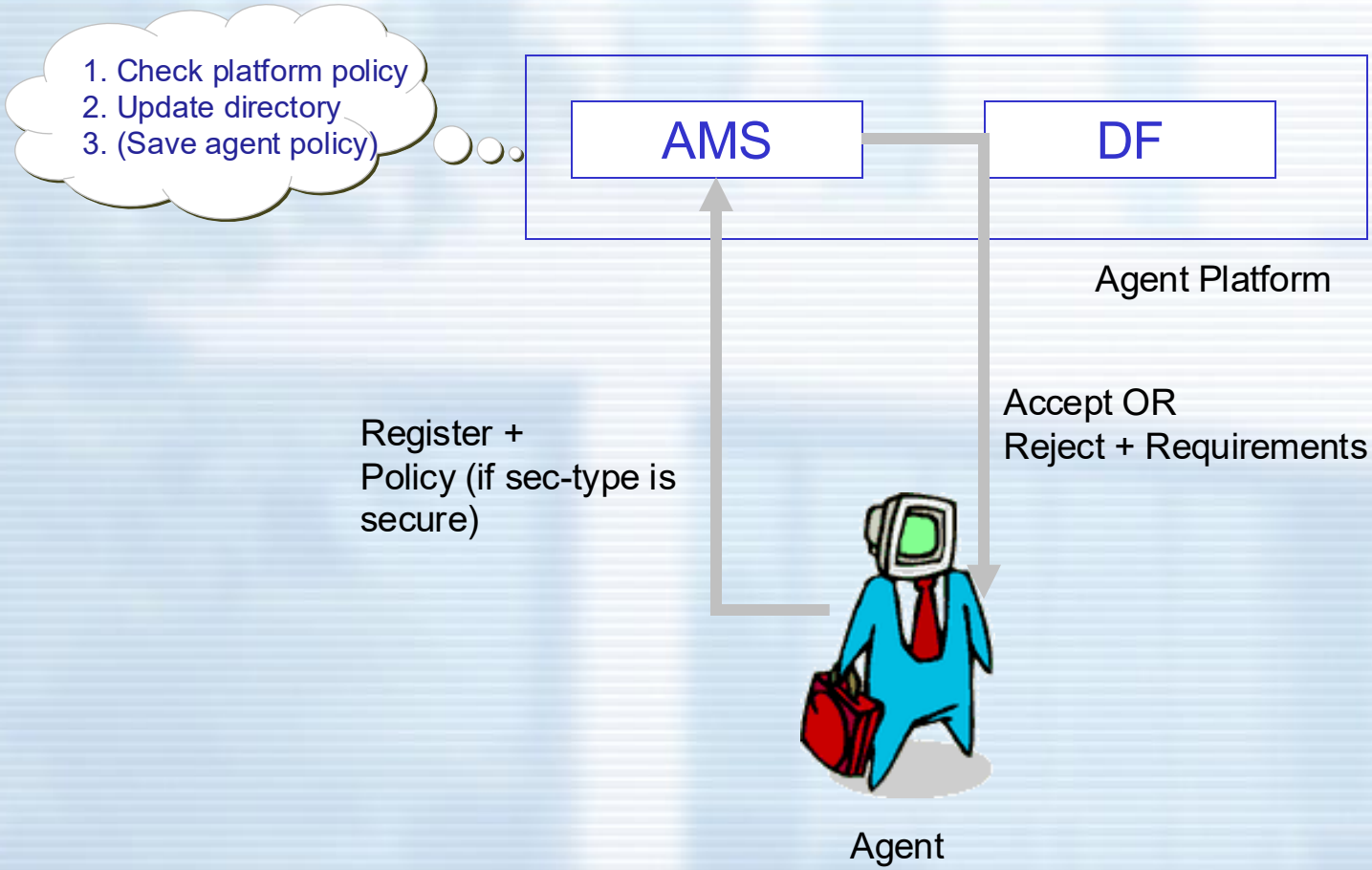
Reject OR
Redirect to credentials
page + policy requirements

http://www…../page.html +
Rei policy in RDF/XML

Framework for agents

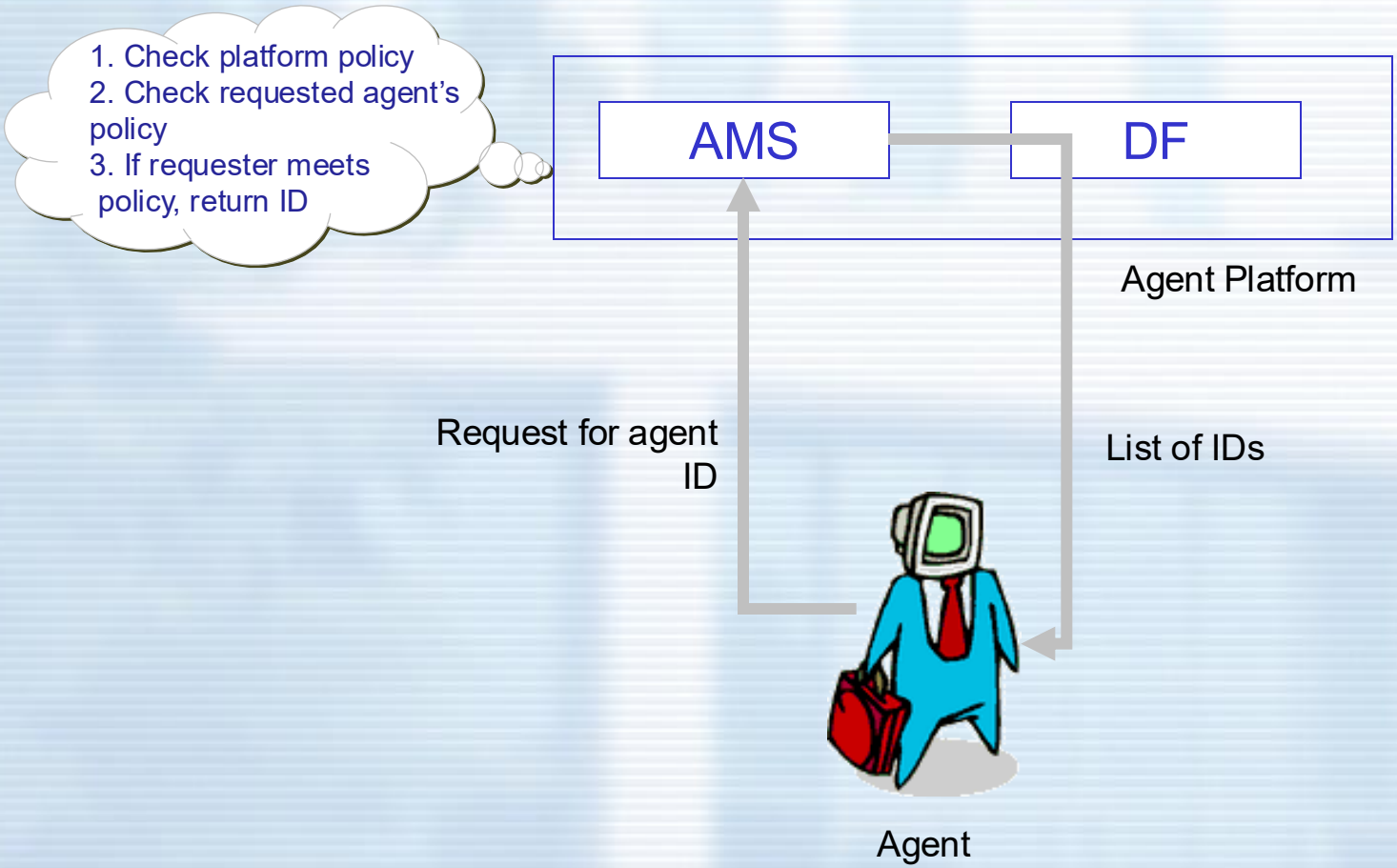
- Framework based on FIPA specs
  - Agents exist on platforms that provide middleware functionality
    - AMS : Agent Management System (white page service)
    - DF : Directory Facilitator (yellow page service)
    - Main functions : registration and querying
- Two levels of security
  - Platform
    - AMS and DF use the platform policy and other policies to decide whether to provide services to the requesting agent
  - Agent
    - Agent uses its own policy to decide whether to honor requests from the platform or other agents

Security Module for AMS



Registration

Security Module for AMS



Querying

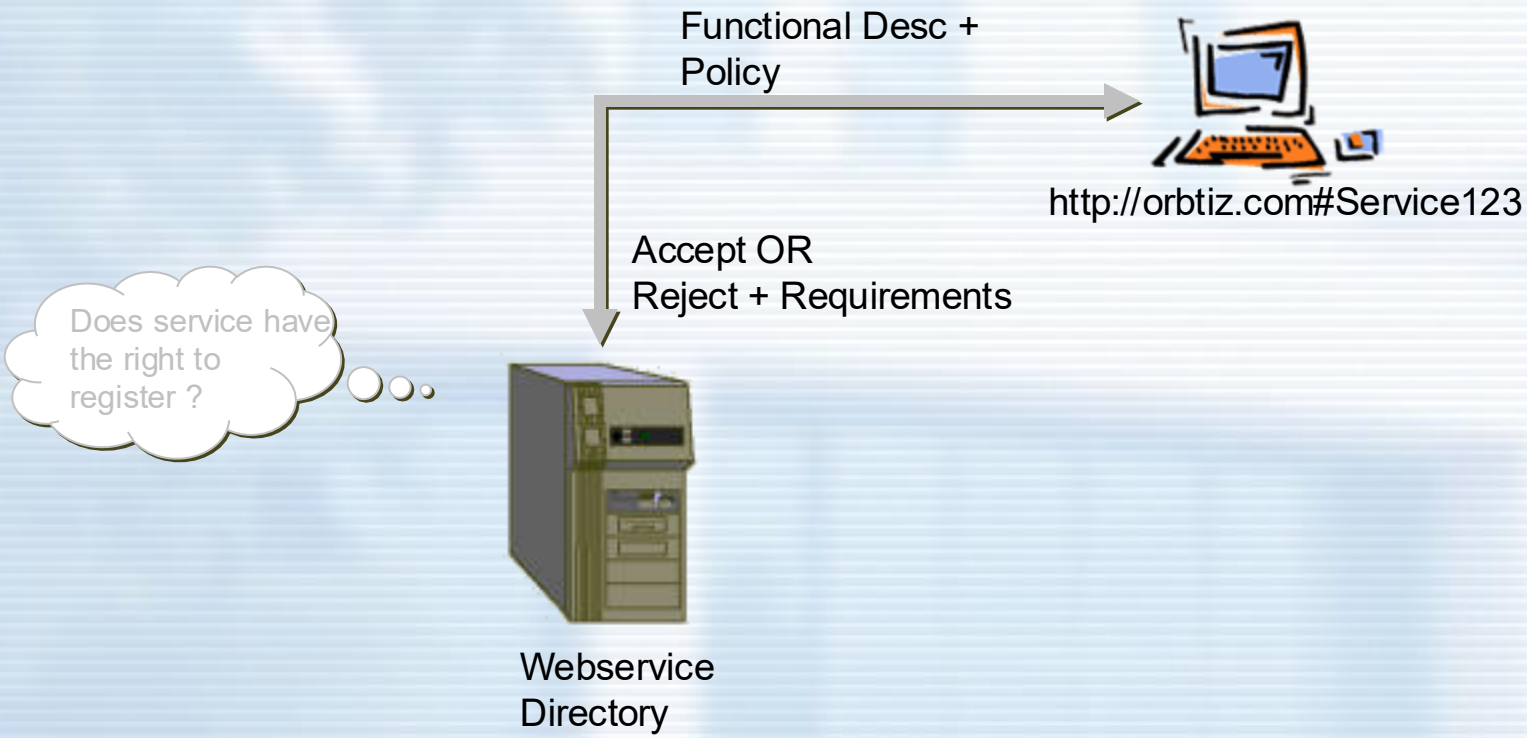
Security Module of DF

- Similar to that of AMS
- Functionality
  - Register a service
    - Checks if agent meets platform's policy for registering a service
  - Query for a service
    - Checks if agent meets the platform's policy for querying for services
    - Finds all matching services (either open or secure)
    - Retrieves associated policies of services registered as secure
    - Returns all open services and those secure services whose policy requirements the requester meets

Agent security

- Security module in the agent is optional
- An agent can rely on the platform to provide authorization to its services
- May have additional policy requirements after initial filtering by AMS and DF

Framework for web services



Registration

Framework for web services

http://orbtiz.com#Service123

1. Does requester have the right to query ?
2. Check that requester meets policy of matched service

Webservice Directory

Request = Func desc of service + Credentials

Reject + Requirements

List of (func + policy) matched services

Query

Example Policy 1

- Service123, of orbitz's namespace, permits users who are in the same current project as an orbitz's platinum club member to use it

Logic

Right(User, service123, Constraints).

Constraints =

currentProject(User,Project),
currentProject(SomeUser, Project),
member(SomeUser, orbitz-platinumClub)

Rei Example Policy 1

```
:x a rei:Variable.
:y a rei:Variable.
:p a rei:Variable.
:R a rei:Right;
    rei:agent rei:x;
    rei:action [a orbitz:findtickets;
          rei:target orbitz:Service123].

:ws-policy a rei:Policy;
rei:grants[a rei:granting;
    rei:to x;
    rei:deontic R;
    rei:oncondition [a
    rei:AndCondition;
```

```
rei:First [a rei:SimpleCondition;
        rei:subject y;
        rei:predicate orbitz:member;
        rei:object orbitz:platinumclub];
rei:Second[a rei:AndCondition;
        rei:First[a rei:SimpleCondition;
                rei:subject y;
                rei:predicate
                  foaf:currentproject;
                rei:object p];
        rei:Second[a rei:SimpleCondition;
                rei:subject x;
                rei:predicate
                     foaf:currentproject;
                rei:object p]]]].
```

Example Policy 2

- All graduate students have the right to delegate a printing action on the HPPrinter in UMBC to any undergraduate student

Logic

Right(Grad, delegate(Grad, UnderGrad, right(UnderGrad, print(UnderGrad, umbc-hpprinter, _, _)_), _), Constraints).

Constraints =

student(Grad, graduateStudent),

student(UnderGrad, undergraduateStudent)

Rei Example Policy 2

```
:s a rei:Variable.
:r a rei:Variable.
:R a rei:Right;
    rei:agent rei:s;
    rei:action [a rei:Delegate;
        rei:Sender s; rei:Receiver r;
        rei:Content [ a
        univ:PrintingAction;
        rei:target umbc:HPPrinter];
    rei:constraints[a
    rei:SimpleCondition;
        rei:subject r;
        rei:predicate rdf:type;
        rei:object
    univ:UndergradStudent].
```

```
:policy a rei:Policy;
rei:grants [a rei:granting;
    rei:to s;
    rei:deontic R;
    rei:oncondition [a
    rei:SimpleCondition;
        rei:subject s;
        rei:predicate rdf:type;
        rei:object univ:GradStudent]
```

Testbeds

- **The past:** Rei's ancestor was used in
  - The EECOMS supply chain management project to control access to information between enterprises
  - The Vigil pervasive computing framework to control access to pervasive services
- **The present:** Rei is currently being used in
  - An agent-based collaboration application (GENOA II) to control team formation and information access
  - The Fujitsu Task Computing framework to control access to pervasive services
- **The future:** Rei will be used in
  - The CoBrA pervasive computing system for privacy policies

Future Work

- Reimplementation in F-OWL
  - We are in the process of reimplementing Rei using the F-OWL reasoning system
- Incorporating OWL rules
  - We hope to use OWL rules in the RDF syntax for Rei if a consensus proposal appears soon
- Reasoning about policies
  - We are extending the reasoner to be able to detect more inconsistencies in policies
- The Rei policy editor
  - We are developing an IDE for Rei policies using the Eclipse framework

Summary

- Security Framework
  - Policy based
  - Distributed
    - Every entity is responsible for its own policy
    - Use of speech acts to modify policies
    - Security is either part of the central directory or controlled by the individual web entity
  - Similar framework for all entities
- Policy Language
  - Based on RDFS + logic
  - Speech acts are tightly coupled with the policies
  - Mechanisms for conflict detection and resolution
  - Can be used for security, management, privacy policies

For More Information

http://rei.umbc.edu/