An Overview of
“Authorization and Privacy for Semantic Web Services”
By L. Kagel, M. Paolucci, N. Srinivasan, G. Denker, T. Finin, K. Sycara

Presented By: Yvette Yoder
Department of Computer Science
Western Michigan University
Instructor: Prof. Leszek T. Lilien
30-minute project-related presentation on October 10, 2006
Main Points
“Authorization and Privacy for Semantic Web Services”

1) Describes an ontology (OWL-S) to annotate input and output parameters to allow for security, encryption, and digital signatures in markup.

2) Discuss a way to add additional security and privacy parameter annotations using the Rei language.

3) Examine an algorithm to check policy compliance--integrated into OWL-S Matchmaker service selection.
Web Service Communication

- We need semantically rich annotations in the descriptions of web services.

- These annotations will assist with the Discovery and Selection of Web Services.
Recall: Semantic Web Architecture

[Saleh, 2001]
Recall: Semantic Web Architecture

- **Layer 3: RDF**
  - Resource Description Framework: Semantics
  - Defines relationships between tagged data resources.

- **Layer 4: Ontology Vocabulary**
  - Description of Terms, Interrelationships of Terms.
  - Expands on RDF to include more detailed properties.

- **Digital Signature (vertical layer):**
  - Added to RDF document to authenticate source
Web Service Discovery

- Services that provide Functional Description + Security Criteria.

- Consider: Web Service A & B:
  - A can perform encryption
  - A requires B to authenticate self
  - A requires B to communicate in XML
  - Framework for the services to communicate.

- OWL-S: Describe details of our Web Services (capabilities & security)

- Matchmaker performs bi-directional matching of services.
Policy = Web Service Description

- Policies specify:
  - Authorization: Who can use a service?
    • Group membership proven with digital certificate.
  - Privacy: Under what conditions can information be exchanged?
    • Encrypted data during transmission
  - Privacy: How can information be used later?
    • State distribution policy of data after receipt.
    • The policy represents a legally-enforceable contract with Web Service.
Services Example: 
The Researcher & The Data Computation Service

- Premise: A Researcher is looking for an online computing service to process experimental data.

- Researcher Privacy Policy Requirements:
  - Requester Policy 1: Wants to send only encrypted data to service.
  - Requester Policy 2: Does not want release of Personal Information to other services or agents.

- DCS Authentication Policy Requirements:
  - Provider Policy 2: Only accepts requests from registered members of a particular organization
Policy Representations:
The Researcher: Representing class Person

<foaf:Person rdf:ID="MarySmith">
  <foaf:name xml:lang="en">Mary Smith</foaf:name>
  <foaf:title>Dr.</foaf:title>
</foaf:Person>

[Kagal et al., 2004]

NOTE: FOAF ontology used to specify domain-specific information about person
Requester Policy 1: Representation-Security:
The Researcher: Wants to send only encrypted data to service.

OWL-S

Namespace abbreviation for OWL-S specification

Looking for service with input parameter ‘PersonInf’ object

Object is of type Encrypted Person Info Object

Constraint: Input contents restricted to structure of class Person, encrypted (EncInfObject is child of BaseObject)

[Kagal et al., 2004]
Provider Policy 1 Representation-Security: The Web Service: Only accepts requests from registered members of a particular organization (using Digital Signature for Authenticated Sign-In).

OWL-S

Looking for service with Registration Info input

Of type Signed Registration Info Object

Constraint: Input contents restricted to structure of class Person, signed. (SigInfObject child of Base Object)

[Kagal et al., 2004]
So Far…

■ Requester: Requires encrypted person information as input to provider’s service.

■ Provider: Requires client to verify registration in a group by providing a digital signature.

■ **Agree**: Both require person information.
Rei

- RDF-S logic based language with an ontology for policy specifications.
- Describes classes and properties, so we can define rules and constraints for policies.
- Modeled on the concepts of rights, prohibitions, obligations.
Rei: Requester Policy 1 - Privacy:

“Any shared personal information must be encrypted.“

[Rei Privacy Policy Right]

Any service that takes input

must take an encrypted person information object.

[Rei: serviceVar a rei:Variable. :someVar a rei:Variable. :PrivPolicy2 a rei:Right; rei:action serviceVar;
  rei:constraint [a rei:AndCondition;
  rei:first[a rei:AndCondition;
  rei:subject serviceVar;
  rei:predicate process:hasInput;
  rei:object someVar];

  rei:second[a rei:SimpleCondition;
  rei:subject someVar;
  rei:predicate process:parameterType;
  rei:object inf: EncPersonInfObj]].

[Kagal et al., 2004]
Rei: Requester Policy 2-Privacy:

“Any service that outputs a Person object shall not be interacted with.”

[Rei Privacy Policy]

Prohibition

Any service that has output

Where that output is simply a FOAF Person object.

Kagal et al., 2004
Rei: Provider Policy 1-Authorization:

“Permit everyone to access the service who is in the same group as the service provider.”

[Kagal et al., 2004]
Rei: Policy Prioritization Capability

- Conflict:
  - Policy1: Requester does not want to share personal information.
  - Policy2: Requester will share information only if it is encrypted.
- Rei can be used to resolve policy conflicts
- Requester can state which policy has priority.
OWL-S policyEnforced Security Property

- Web Service Authorization Policy:

  ```xml
  <profile:Profile rdf:ID="DataComputationService_Profile">
    <profile:textDescription>
      This data computation service requires authorization.
    </profile:textDescription>
    ...
    <policyEnforced rdf:resource="#AuthPolicy1"/>
  </profile:Profile>
  
  Provider service requires Authorization Policy 1 be enforced.
  ```

- Web Service Privacy Policy:

  ```xml
  <foaf:Person rdf:ID="MarySmith">
    <foaf:name xml:lang="en">Mary Smith</foaf:name>
    <foaf:title>Dr.</foaf:title>
    <policyEnforced rdf:resource="#PrivPolicy1"/>
    <policyEnforced rdf:resource="#PrivPolicy2"/>
  </foaf:Person>
  
  Requester service requires Privacy Policies 1 & 2 be enforced.
  ```

[Kagal et al., 2004]
Using Policies for Service Selection

- Requester needs to verify compatibility of its policies with the policies of the provider specified in OWL and Rei.
- Rei allows for reasoning on policies to evaluate compatibility of right, prohibitions, etc.
- OWL-S integrated with a capability–based matching engine: MatchMaker
Matchmaker Policy Compatibility Algorithm

Step 1) Matchmaker selects a provider with capabilities matching requester.

Step 2) Matchmaker extracts policies of both.

Step 3) Matchmaker uses Rei reasoning to evaluate compatibility.

- If incompatible -> provider abandoned; continue to check next service
- Else provider selected
What happens if the Provider is not Honest?

- Provider encouraged to be honest and explicit in specifying and enforcing policies:

- Otherwise:
  - **May Lose Business**: Some requesters may not want to adhere to policies stated (even if they don’t know the policies aren’t being enforced).
  - **May Lose Requester Trust**: If they realize policies are not being enforced.
  - **Transactions Will Fail**: If Provider does not explicitly specify its policies, yet tries to interact with a client not accepting those policies.
Relation to Oppnets

- Policies can be explicitly provided via annotated markup.
- OWL-S or a similar ontology could be used to specify the existence of node privacy and authorization policies.
- Rei or a similar language could be used to extend an ontology and define specific policy constraints for node selection and interaction.
References
