Semantic Policy Enforcement and Reconciliation for Information Exchange in XMPP

VIStology, Inc.

Brian Ulicny, Won Ng, Oleg Simakoff, Jakub Moskal, Mitch Kokar (Northeastern)

STIDS 2011

Outline of Presentation

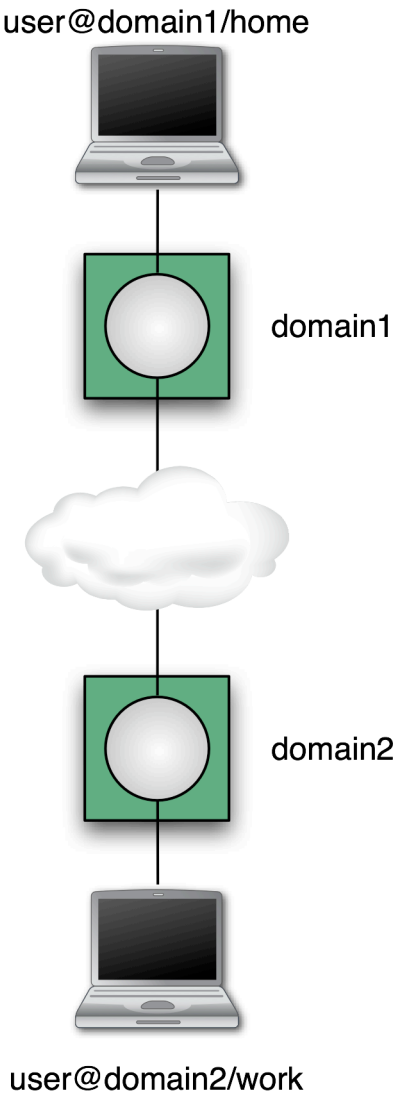
- About VIStology
- Problem outline
  - XMPP
- Policy Representation
  - SBVR, BaseVISor, Ontology
  - Authoring and Editing
  - Metamodel-based Translation
- Information Exchange Use Cases
  - Presence based on Attributes
  - Security Levels
  - Policy-governed chat
  - Ontology-matching
- Wrap up

About VIStology, Inc.

- Past and present R&D contracts
  - ONR, Army RDECOM , AFRL/AFOSR, DARPA, MDA
- Professional Collaborations
  - Northeastern University, W3C, Lockheed Martin, OMG, Referentia Systems
- Products & Services
  - BaseVISor: highly efficient inference engine
  - ConsVISor: consistency checker of ontologies
  - PolVISor: Policy Reconciliation Framework
  - Ontology Engineering and Systems Design
- Areas of Expertise
  - Level 2+ Information Fusion, Situation Awareness, Formal Reasoning Systems, Policy Based Control, Ontology Engineering, Information Retrieval, NLP

XMPP Architecture

- XMPP - eXtensible Messaging and Presence Protocol
- Servers talk to other servers
  - DNS lookup on domain portion of address
  - Dialback, MTLS for security
  - One connection for many conversations
- Client talks to “local” server
  - Wherever the user account is hosted
  - Tied to directory if desired
- Addressing Scheme:
  - JID = Jabber ID = `node@domain/resource`
  - node: identity, e.g. user name
  - domain: DNS domain name
  - resource: device identifier
  - `node@domain` identifies a user
  - `resource` identifies a nick name



Stanzas

- Any action in XMPP communicated as a stanza
- All have `to='JID'` and `from='JID'` addresses
  - `to` gives destination
  - `from` added by local server
- Each stanza routed separately
- All contents of stanza passed along
- In XML format
- Different types for delivery semantics

  `<message/>`: one direction, one recipient
  `<presence/>`: one direction, publish to many
  `<iq/>`: "info/query", request/response

- In our implementation, XML stanzas are translated from XML to RDF (using XSLT) for reasoning by PolVISor

Military Use of XMPP Chat

- Used for a lot of things in the military:
  - Navy bridge-to-bridge comms
  - Logistics support
  - Mission planning
- In theory it's not an official command & control link - in reality it is
- Grew from the bottom up; individual commands and organizations "unofficially" deployed it
- There was no initial standard or design effort; lots of incompatible systems out there
- There may be other problems:
  - unaudited code, no authentication of users/roles
  - Multi-level security, coalition security, etc.
- XMPP has been adopted as the mandatory standard for chat by the DoD IT Standards Registry
- No other chat protocol has been approved

Problem Statement

We are concerned with the ability to use policies to **ensure compliance during runtime** as well as with the ability to **do policy reconciliation in the context of information exchange in XMPP**

Policy compliance involves the run-time process of ensuring that all of the conditions defined by a policy hold true; e.g., controlling presence visibility according to server's policies

In policy reconciliation, the goal is to take multiple polices and generate a session policy instance that simultaneously satisfies all of them; e.g., admitting a user to a chatroom based on policies of multiple servers

Expressing Security Policies in Restricted Natural Language

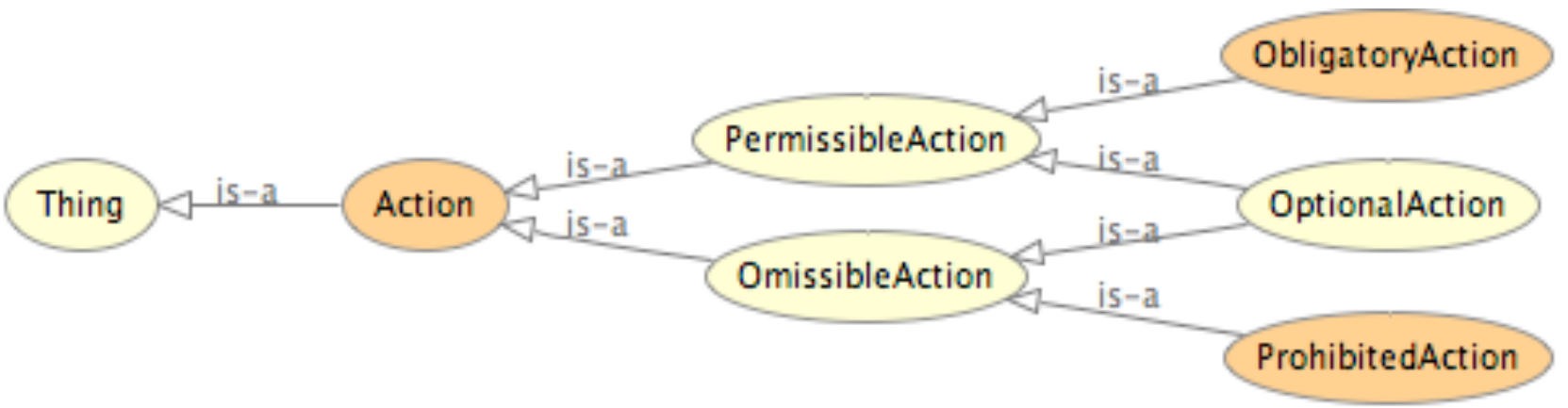
- PolVISor uses policies expressed in **SBVR** Structured English
  - An OMG Standard
  - Semantics of Business Vocabulary and Rules
  - Provides constructs for representing modalities:
    - Deontic: "It is obligatory/permitted/forbidden that ..."
    - Alethic: "It is necessary/possible/impossible that …"
  - Has corresponding XML (XMI) representation of logical structure
  - First-order quantification (some/a, all/each/every)
  - Non-standard coindexing:
    - "a boy shaved the boy" means "a boy shaved himself"
  - N-ary predicates represented as "fact types"
    - Binary: read(actor,file), write(), delete()
    - Ternary: send_to(sender,message,recepient), receive_from(), etc

Expressing Security Policies in Restricted Natural Language

- Color/font-coded syntax
  - term : noun concepts (other than individual concepts), e.g. adult
  - Name : individual concepts, e.g. John
  - *verb* : fact types, e.g. *reads*
  - keyword : reserved words or phrases with special meaning in SBVR, e.g. each, it is obligatory that, the, a, and, …

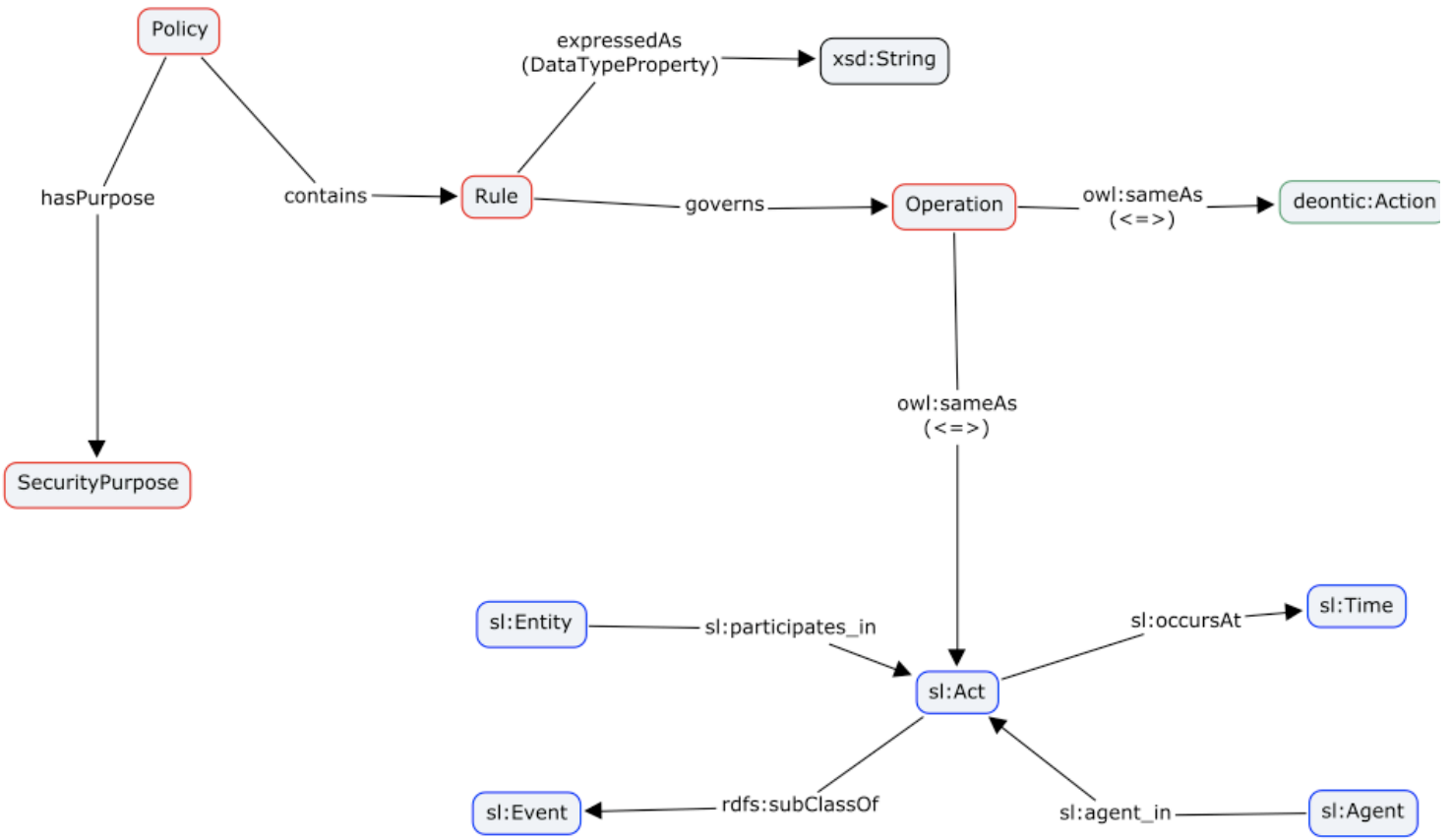
- It is prohibited that
  a person that *is* not an adult *reads* a pictureWithMatureContent

Deontic Action Ontology



Expressed in OWL using disjointClass, subClassOf, complementOf, intersectionOf…

SecPol Ontology



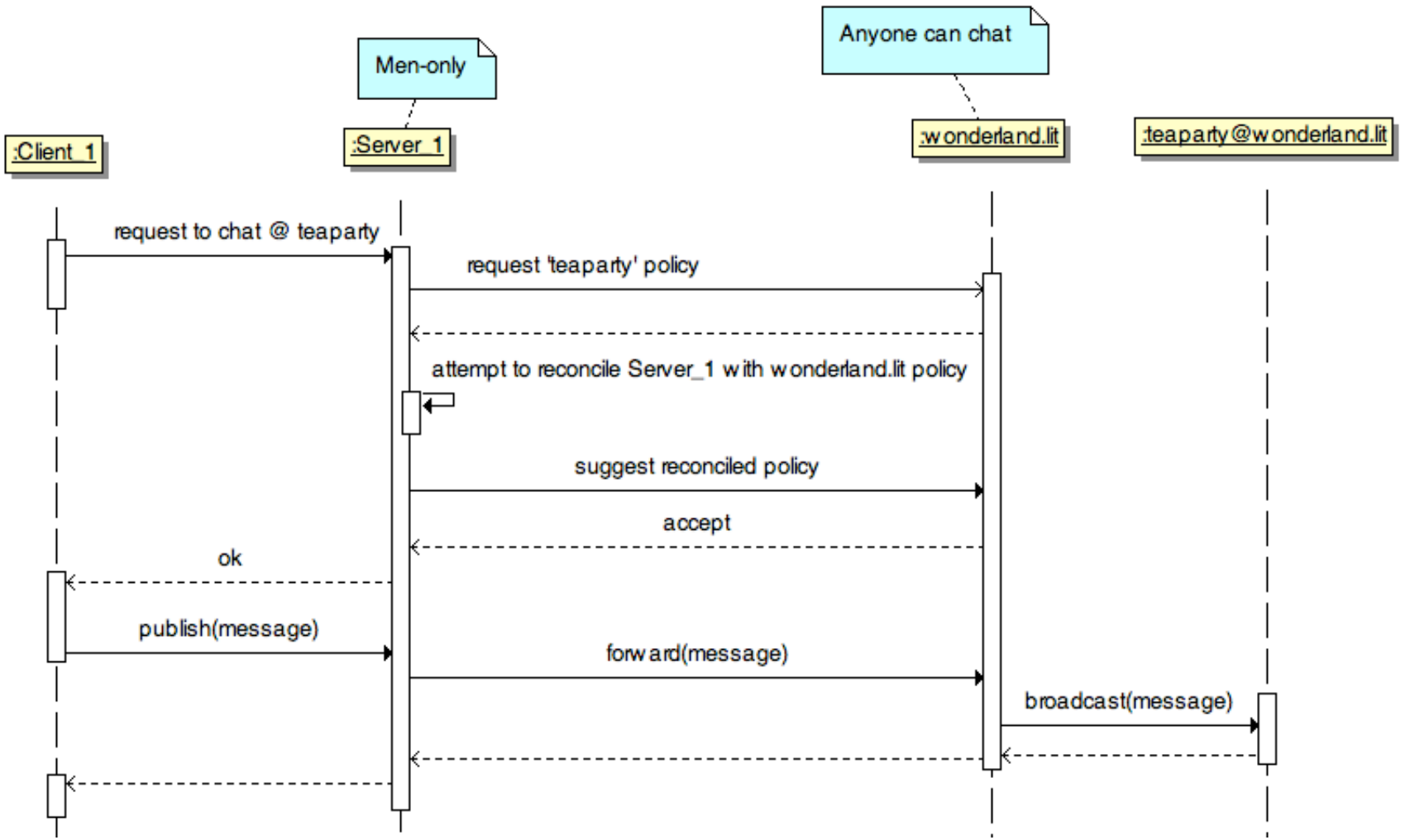
PolVISor: Policy Editing-Translation-Execution

- Policy Rules are written in a subset of Semantics of Business Vocabulary and Rules (SBVR) Structured English (SE)
  - A rule author uses the PolVISor editor to enter the SBVR vocabulary corresponding to the domain ontology to support the rules
  - Then the editor writes rules in that vocabulary
- After converting the SBVR SE rules to XMI, PolVISor can translate the XMI into BaseVISor rule language (BVR) via ATL scripts
  - ATL is a transformation language and execution environment that converts a source model conforming to a source metamodel (e.g. SBVR) to a target model conforming to a target metamodel (e.g. BVR).
- After converting the SBVR XMI to BVR, the resulting BVR rules can be executed by the BaseVISor OWL 2 RL reasoner

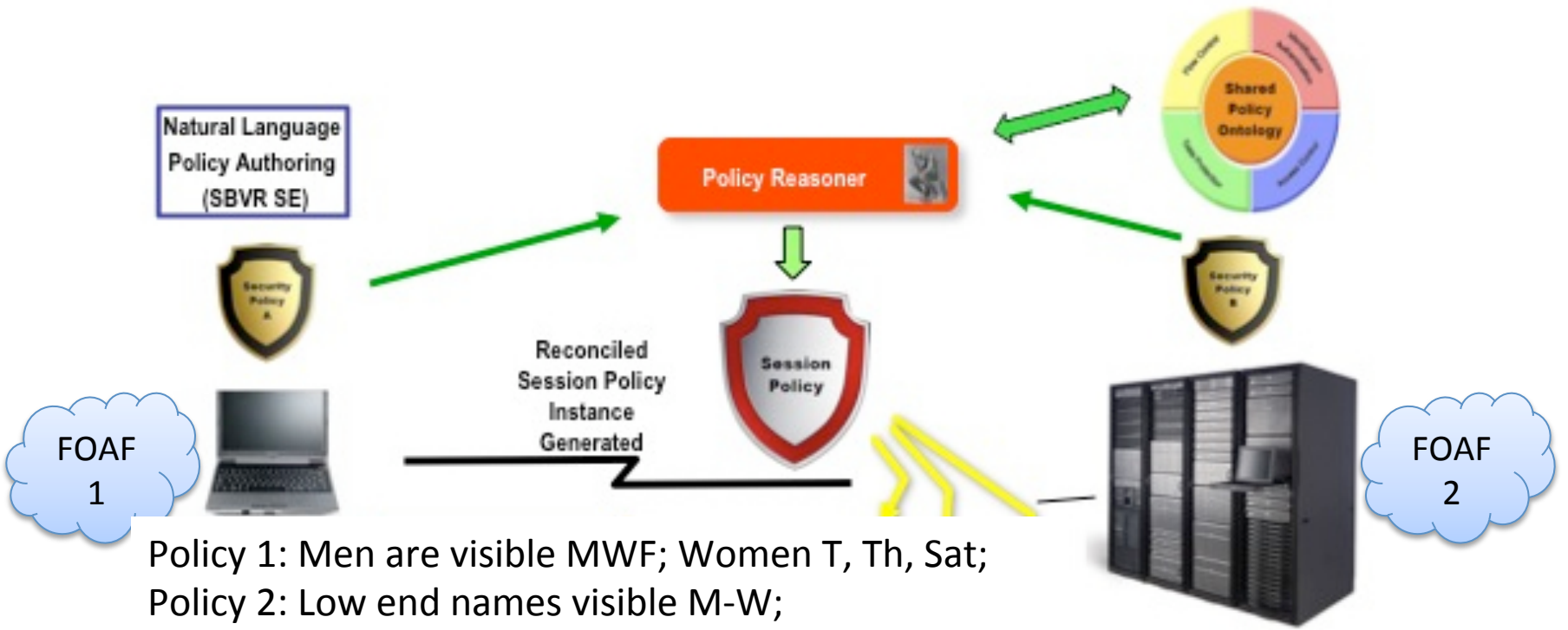
XMPP Implicit/Explicit Reconciliation

- Clients can chat with other clients who are online. A client's online status is indicated by sending a "presence" packet to his contacts.
- Then packets can be sent to those present, provided delivery polices are satisfied
- Standard XMPP allows clients to restrict or block contact with others, but does not allow servers to limit contact.
- With PolVISor, a server can specify policies to allow or restrict chats between clients.
- Our plugin to the OpenFire XMPP server intercepts packets, including presence and message packets and forwards them to PolVISor.
- If the contact is on a different server, the server forwards the packet to the server of the contact.
- If the contact is on the same server, the server forwards the packet to the contact.
- Each server with a plug-in does a compliancy check that the packet is allowed by its authorization rules.
- The presence is delivered if:
  - an instance of at least one sender server policy permits the request and
  - an instance of at least one recipient server policy permits the request and
  - no instance of any policy denies the request.
- Otherwise policies are exchanged and explicit reconciliation is performed

Policy Reconciliation Sequence Diagram



Policy-Based XMPP Presence Exchange

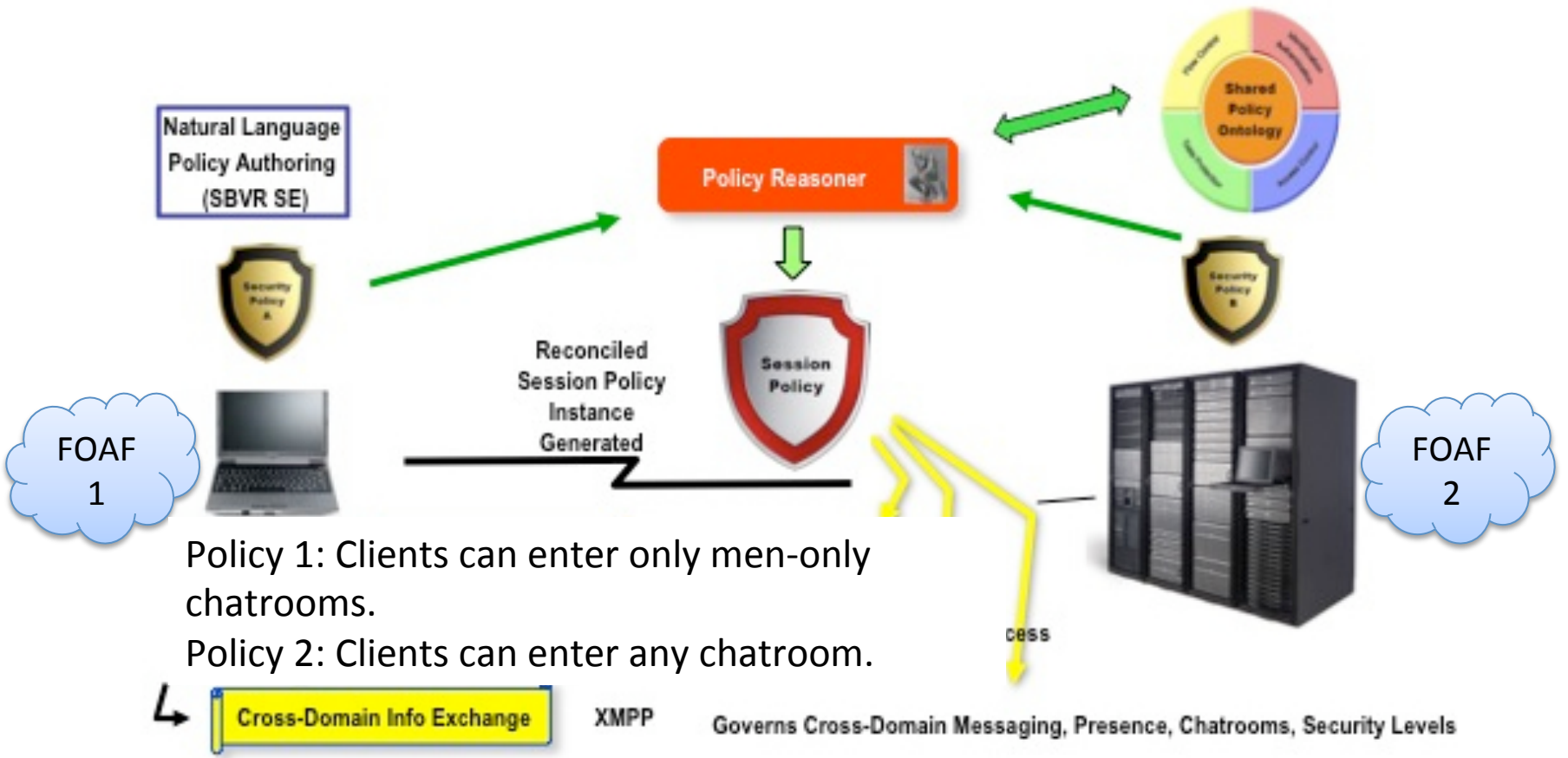


Policy 1: Men are visible MWF; Women T, Th, Sat;
Policy 2: Low end names visible M-W;
High end Th -Sat

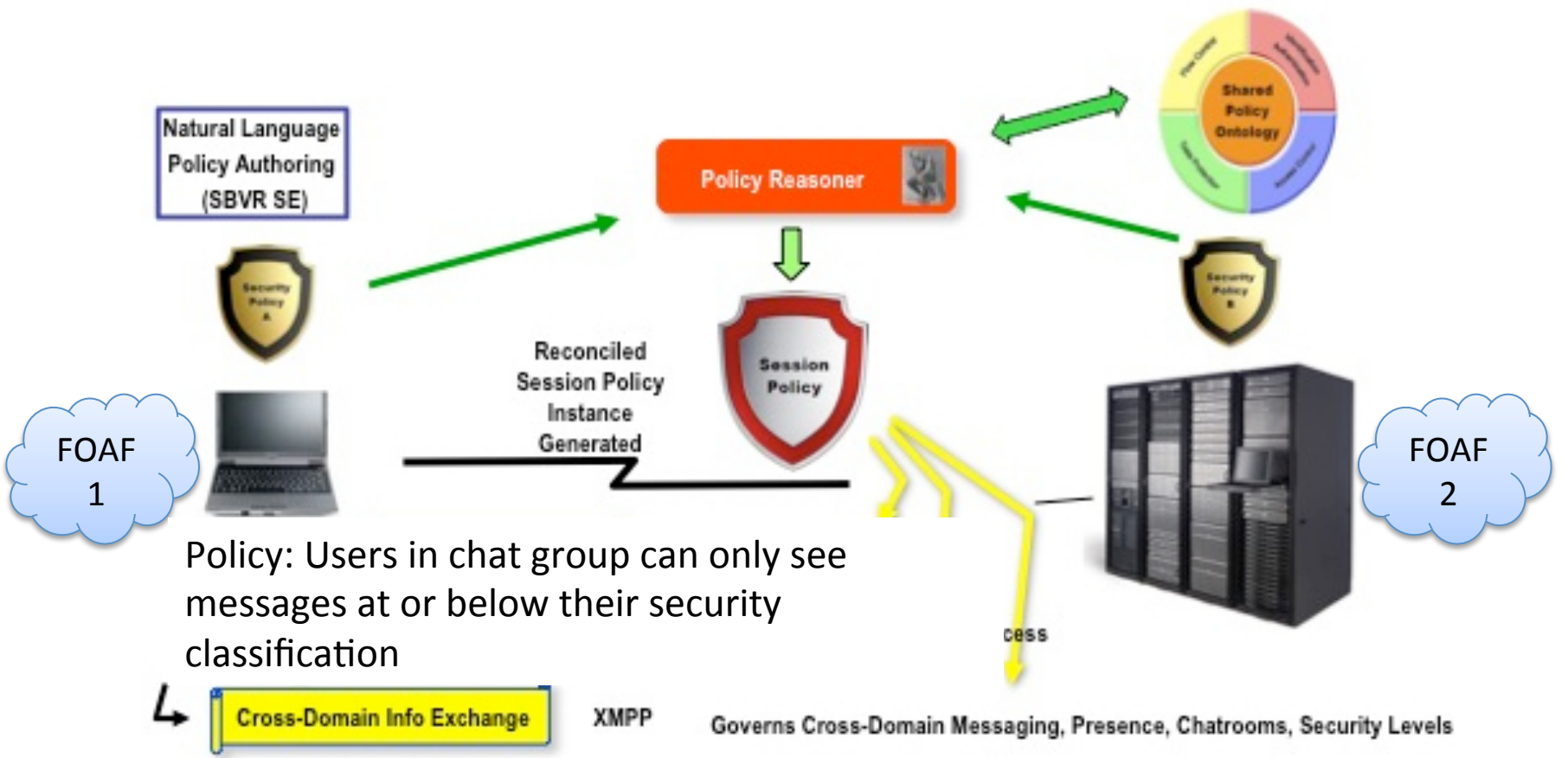
Cross-Domain Info Exchange XMPP Governs Cross-Domain Messaging, Presence, Chatrooms, Security Levels

Can't be implemented in standard XMPP.

Policy Based XMPP Chatrooms



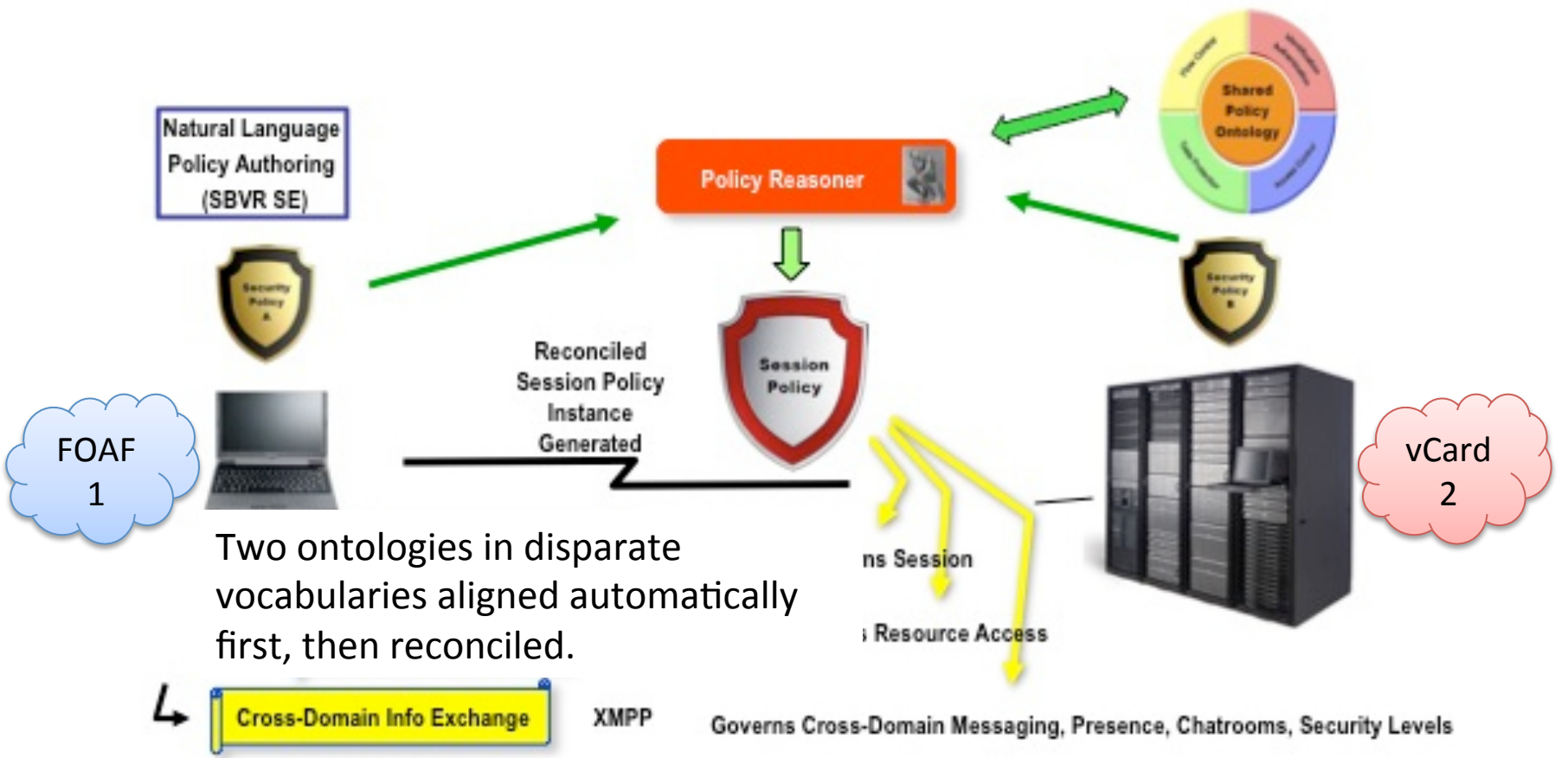
Security Marking-Based XMPP Info Exchange



IC-ISM Ontology

- ISM Ontology v 0.7
- By Richard Lee, Booz Allen Hamilton, lee_richard@bah.com
- A rendering of the IC-ISM XML spec for security markings. It is based on the IC-ISM v5 XSD, updated thru 2010-09-25.
- Ontology is intended to use OWL 2 annotations to associate triples with classification
- Need not be used this way, however.
- Provides lists of allowed values (CVEs) for (virtually) all the CAPCO-specified parameters; these are in the CVE(_ISM) ontology.
- Since this ontology perforce uses properties allows multiple instances. Note that In most cases this also entails dropping the "s" from the name.
- 65 Classes
- 17 Object Properties
- 20 Data Properties
- 4220 Individuals

Policy-Based XMPP Info Exchange with Ontology Matching



Summary

Demonstrated:

- Representation of policies in SBVR Structured English
- Automated conversion of policies to executable OWL 2 RL and BaseVISor rules (BVR)
- Enforcement of policies using ontology and rule based reasoning
- Exchange and automatic reconciliation of policies of different nodes
- Matching of policies written in different ontologies
- Application of all of the above in the XMPP chat scenarios

Demo videos at http://173.14.188.57:9999/secpol/

Backup Slides

XMPP Classified Messages: Implicit Reconciliation

- A client can join chatrooms and send group chat messages to each client in the chatroom.
- Standard XMPP allows clients to restrict or block contact with others, but does not allow servers to limit contact.
- With PolVISor, a server can specify policies to allow or restrict chats between clients.
  - Servers allow messages marked with a security label (e.g. Restricted) if the sender has a sufficient clearance level (e.g. Secret).
  - Servers allow messages marked with a security label (e.g. Restricted) if the recipient has a sufficient clearance level (e.g. TopSecret).
  - All other messages marked with a security label are denied and dropped.
- A server plug-in intercepts packets, including presence and message packets and forwards them to PolVISor.
- If the packet is allowed by PolVISor, the server processes the packet as usual
  - If the contact is on a different server, the server forwards the packet to the server of the contact (or chatroom)
  - If the contact is on the same server, the server forwards the packet to the contact (or chatroom)
- Each server with a plug-in does a compliancy check that the packet is allowed by its authorization rules.
- If a client on one server tries to send a marked classified message to a chatroom hosted on a different server and both servers allow the packet, reconciliation is implicit, i.e. if reconciliation is possible, the message is delivered if there is:
  - an instance of the sending server's policy that permits marked messages if the sender has a sufficient clearance level and
  - an instance of the recipient server's policy that permits marked messages if the recipient (in this case, the chatroom) has a sufficient clearance level.

XMPP Chatroom: Explicit Reconciliation

- A client can join chatrooms by sending a "presence" packet to the chatroom.
- Standard XMPP allows clients to restrict or block contact with others, but does not allow servers to limit contact.
- With PolVISor, a server can specify policies to allow or restrict chats between clients.
  - E.g. Server 1 only allows men to join chatrooms. All others (females or contacts with unspecified gender) are not allowed in chatrooms. Men are not allowed in chatrooms that can potentially contain non-men.
  - E.g. Server 2 allows anyone to join chatrooms.
  - If a client on Server 1 wants to join a chatroom on Server 2, to ensure that neither servers' rules are violated (e.g. Server 1's rule that contacts on only join chatrooms that can only contain men), reconciliation must be performed and both servers must enforce this reconciled policy.
- A server plug-in intercepts packets, including presence and message packets and forwards them to PolVISor.
- The client's server (Server 1) requests the chatroom's server's (Server 2) policy and forwards both its own policy and Server 2's policy to PolVISor.
- PolVISor returns a reconciled policy by AND'ing the conditions of both policies, i.e. performs explicit reconciliation.
- Server 1 enforces this policy and forwards the reconciled policy to Server 2, who must also enforce this policy.
- Each server with a plug-in does a compliancy check that the packet is allowed by the reconciled policy.
- If the packet is allowed by PolVISor based on the reconciled policy, the server processes the packet as usual.
  - The sending server forwards the presence to the chatroom server.
  - The chatroom server forwards the presence to the chatroom.

XMPP Ontology Matching

- A client can join chatrooms by sending a “presence” packet to the chatroom.
- Standard XMPP allows clients to restrict or block contact with others, but does not allow servers to limit contact.
- With PolVISor, a server can specify policies to allow or restrict chats between clients.
  - E.g. Server 1 only allows men to join chatrooms. All others (females or contacts with unspecified gender) are not allowed in chatrooms. Men are not allowed in chatrooms that can potentially contain non-men.
  - E.g. Server 2 allows anyone to join chatrooms.
- If a client on Server 1 wants to join a chatroom on Server 2, to ensure that neither servers’ rules are violated, reconciliation must be performed and both servers must enforce this reconciled policy.
- Since Server 1 and Server 2 use different ontologies to represent gender of their clients, FOAF and vCard respectively. Before policies can be reconciled, ontologies must be first matched to produce alignment. Alignment can be expressed in terms of axioms that bridge two different ontologies and allow the same reconciled policy to be enforced on both servers
- A server plug-in intercepts packets, including presence and message packets and forwards them to PolVISor.
- The client’s server (Server 1) requests the chatroom’s server’s (Server 2) policy (which includes the vCard ontology) and forwards both its own policy and Server 2’s policy to PolVISor.
- PolVISor uses automated ontology matching to produce alignment between FOAF and vCard ontologies.
- PolVISor returns a reconciled policy by AND’ing the conditions of both policies, i.e. performs explicit reconciliation. The reconciled policy makes use of the FOAF ontology to access client’s gender and includes the alignment axioms that bridge the policy with the vCard ontology.
- Server 1 enforces this policy and forwards the reconciled policy to Server 2, who must also enforce this policy.
- Each server with a plug-in does a compliancy check that the packet is allowed by the reconciled policy.
- If the packet is allowed by PolVISor based on the reconciled policy, the server processes the packet as usual.
  - The sending server forwards the presence to the chatroom server.
  - The chatroom server forwards the presence to the chatroom.
- If a client on Server 2 (with gender represented using the vCard ontology) wants to join a chatroom on Server 2 after the two servers performed policy reconciliation, Server 2 enforces the reconciled policy and makes use of the alignment axioms.