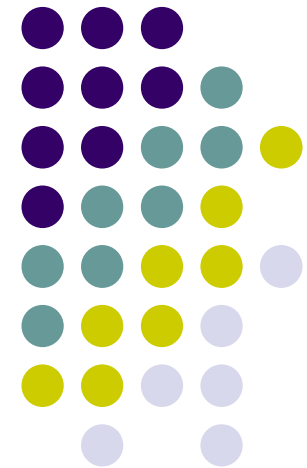


# A Modeling Framework for Generating Security Protocol Specifications

**Genge Béla** and Haller Piroska  
“Petru Maior” University of Târgu Mureș,  
N. Iorga Str., No. 1, Târgu Mureș,  
jud. Mureș, ROMANIA,  
{bgenge, phaller}@upm.ro





# Outline

- Introduction
- Existing solutions
- Developed specification
- Modeling framework:
  - Modeling sequential component
  - Modeling semantic component
- Example modeling of a security protocol
- Experimental results
- Conclusions and future work



# Introduction

- Security protocols: “communication protocols in which cryptography is used to achieve security properties” (Cremers et al, 2005)
- Security protocol specifications: “descriptions of protocol participant operations and messages” (Abadi, 1999)
- Web services: “interfaces that describe a collection of operations that are network-accessible through a collection of standard XML messages” (Gottshalk et al, 2002)



# Introduction (cont.)

- Main goals:
  - Automatic execution of security protocol specifications in Web services specific environments
  - Implementing classic security protocols using XML-based technologies
- Based on: security protocol specifications
- Requires detailed description of:
  - Transport protocol
  - Protocol participant roles
  - Message structures
  - Cryptographic parameters
  - Construction and verification operations



# Existing solutions

- SAML (Security Assertions Markup Language): provides well defined protocols for authentication and authorization
- WS-Security (Web Service Security): provides descriptions for implementing security protocol specific message components as XML structures
- Papers related to automated execution:
  - Mengual et all (2002) – based on a formal description
  - Abdullah et all (2003) – based on XML descriptions from which code is generated and which requires re-compiling applications

# Existing solutions (cont.)

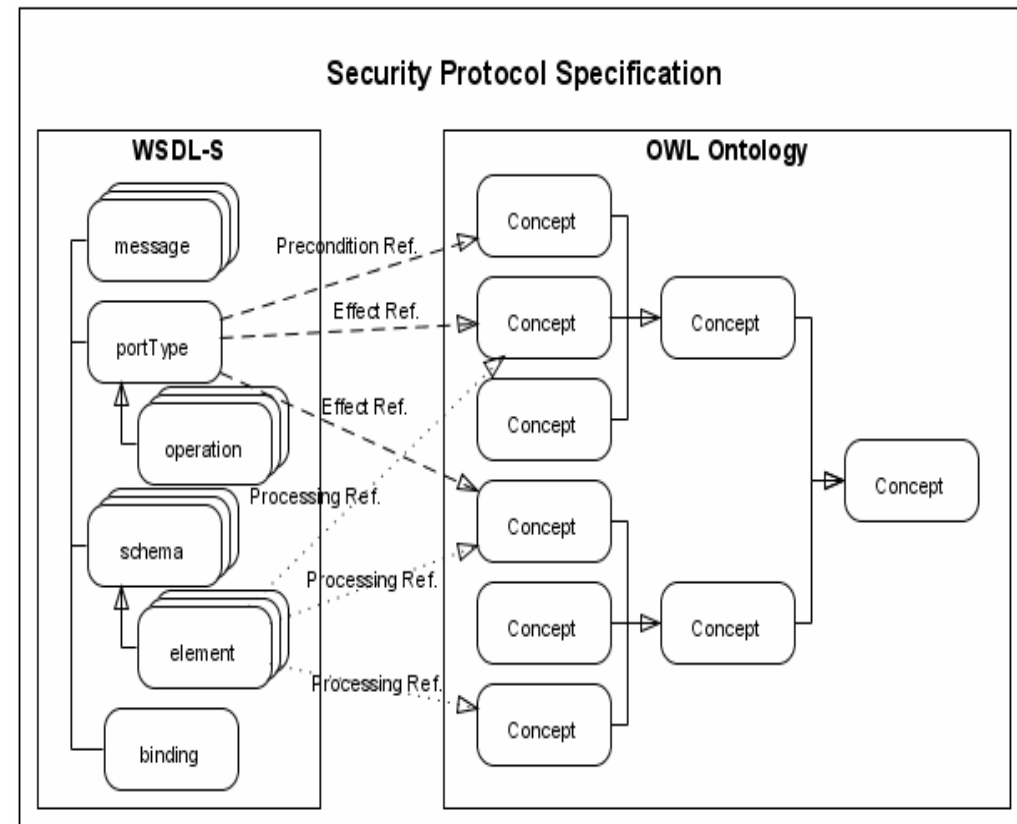


- Shortcomings:
  - Flexibility
  - Extendibility
  - Difficulties in integrating into Web services

# Proposed specification construction



- Based on Web services specific technologies
- Two components:
  - Sequential
  - Semantic





# Sequential component

- Contains information related to:
  - Message transport
  - Message sequences
  - Message component annotations
  - Protocol preconditions and effects
- Implemented using: WSDL-S



# Semantic component

- Contains information related to:
  - Message construction:
    - Encryption algorithm, key
    - Message component location
    - Message component type
  - Message processing:
    - Decryption algorithm, key
    - Message component verification
- Implemented using: OWL



# Modeling framework

- Based on rules for modeling the two components
- Requirements:
  - Extracted from the informal specification:
    - Explicit participant specification
    - Identify message component types and direction
    - Identify cryptographic requirements
  - Processing:
    - Identify initial, generated, discovered and loaded components
    - Identify processing mechanisms such as encryption, decryption, verification or storage



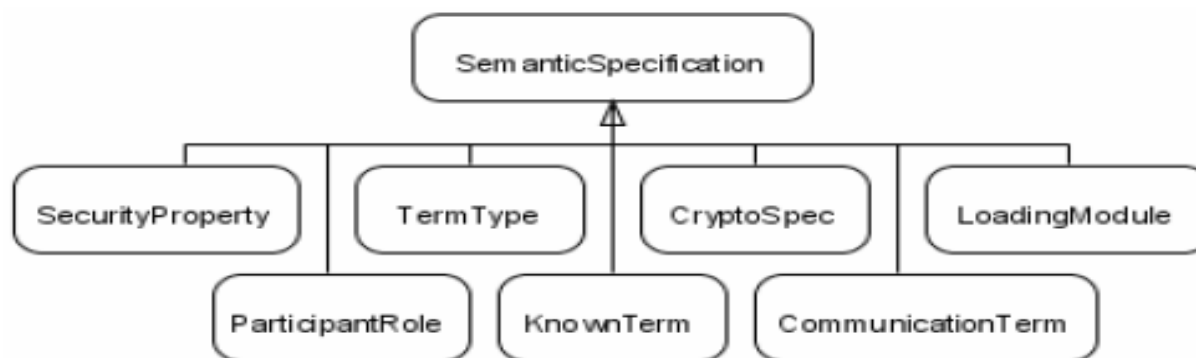
# Modeling framework (cont.)

- Modeling the sequential component:
  - Establish and model preconditions and effects:
    - Random numbers
    - Keys
    - User names
    - Protocol goal: authentication, secrecy, key exchange
  - Establish and model annotated message sequences



# Modeling framework (cont.)

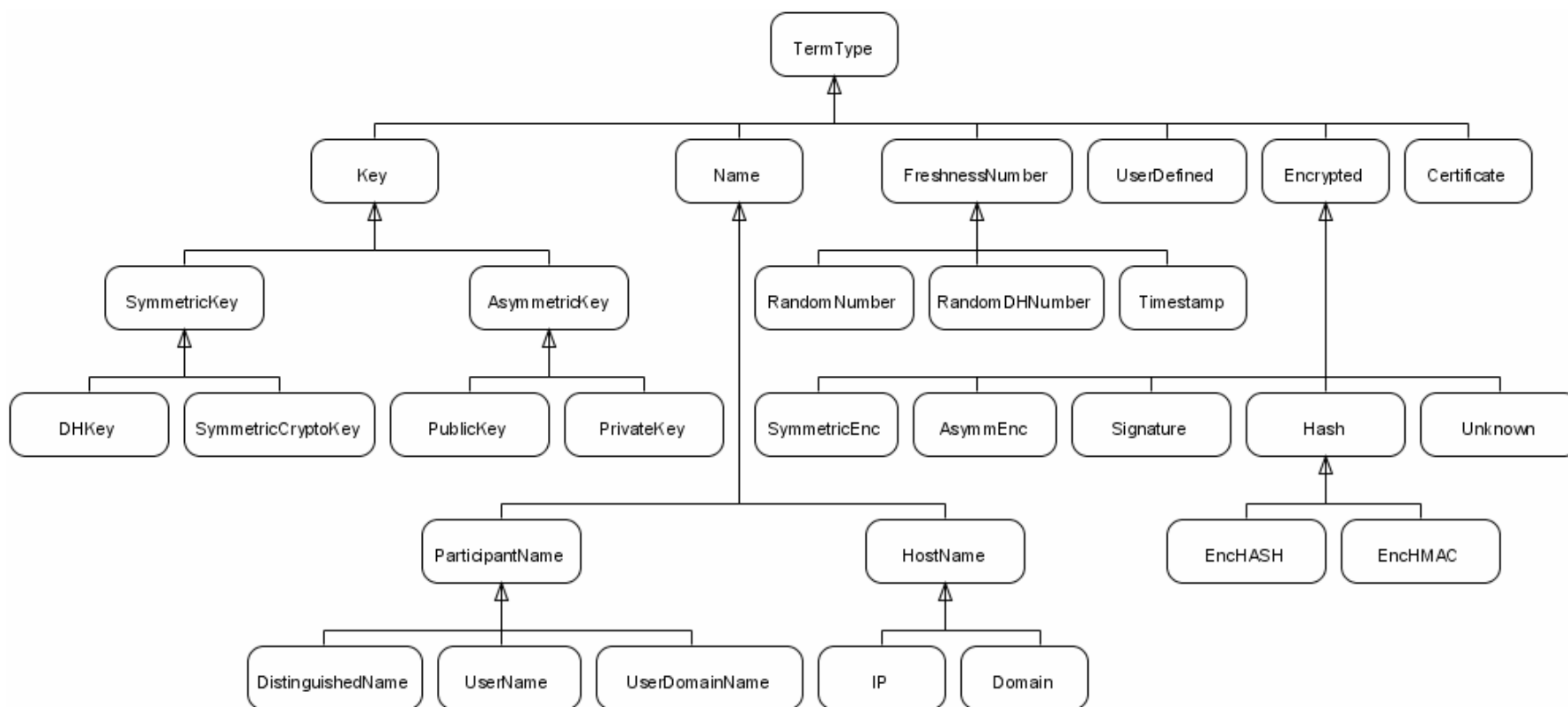
- Modeling the semantic component:
  - Extend the core ontology (KnownTerm and CommunicationTerm sub-ontologies) for each specification with concepts and properties
  - Added concepts correspond to:
    - Participant knowledge: initial, generated, discovered and loaded
    - Communication (sent and received) components
  - Added properties correspond to:
    - Construction and processing operations: encrypt, decrypt, verify, store, etc.
    - Term classification: key, random number, symmetrical, asymmetrical, hash, etc.
    - Module classification: key, random generator, previous knowledge





# Modeling framework (cont.)

- Example static sub-ontology (TermType):



# Modeling framework (cont.)



- Example sequential component modeling rule:

*Let prop be a security property:*

- *If prop = MutualAuthentication then*

$$\exists c_1, c_2, c_3 \in CONC : PREC = PREC \cup \{ \langle uri, c_1 \rangle, \langle uri, c_2 \rangle, \langle uri, c_3 \rangle \}$$

- *Else*

$$\exists c_1, c_2 \in CONC : PREC = PREC \cup \{ \langle uri, c_1 \rangle, \langle uri, c_2 \rangle \}$$



# Modeling framework (cont.)

- Example semantic component modeling rule:

*Let  $c$  be a sub-concept of  $GeneratedTerm$  with the type  $SymmEncrypted$ . Then there exists a  $hasKey$  and a  $hasSymmetricAlgorithm$  property defined.*

$$\begin{aligned} &\forall c \in subcon(GeneratedTerm) \cup \\ &subcon(DiscoveredTerm) \text{ if } \exists p \in prop(c) : \\ &range(p) = SymmEncrypted \text{ then } \exists p', p'' \in prop(c) : \\ &\quad name(p') = hasKey \wedge name(p'') = hasSymmetricAlgorithm \wedge \\ &\quad mincard(p') = maxcard(p') = 1. \end{aligned}$$

# Example specification construction



- “BAN Concrete Andrew Secure RPC” protocol
- A corrected version published by Gavin Lowe in 1997
- For each participant a separate specification is constructed

- Informal specification:

$$A \rightarrow B: A, N_a$$

$$B \rightarrow A: \{N_a, K, B\}_{K_{AB}}$$

$$A \rightarrow B: \{N_a\}_K$$

$$B \rightarrow A: N_b$$

# Example specification construction (cont.)



- Sequential component is generated as a WSDL-S specification
- Initiator role modeled as a precondition
- The goal of the protocol, the exchange of a session key, modeled as an effect
- Messages are modeled as XML schemas

```
...
<complexType name="Msg1Request">
  <sequence>
    <element name="Participant A" type="xsd:string"
      wsse:modelReference="../../../SecProt.owl#Sent_A"/>
    <element name="Random" type="xsd:base64Binary"
      wsse:modelReference="../../../SecProt.owl#Sent_Na"/>
  </sequence>
</complexType>
...
<wsdl:portType name="Encrypted communication">
  <wsdl:operation name="Msg1">
    <wsdl:output message="tns:Msg1Request"/>
  </wsdl:operation>
  ...
  <wsse:precondition name="Initiator"
    wsse:modelReference="../../../SecProt.owl#
      Initiator_role"/>
  <wsse:effect name="SessionKeyExchange"
    wsse:modelReference="../../../SecProt.owl#
      Session_key_exchange"/>
</wsdl:portType>
```

# Example specification construction (cont.)



- Semantic component is generated as an OWL specification
- For each annotation semantic operations are generated





# Experimental results

- Constructed over 38 specifications corresponding to initiator and respondent protocol roles
- Automatically executed by participants in a developed platform aiming at the automated discovery of Web services that use heterogeneous security protocols

Protocol participant	Specification processing (ms)	Message construction (ms)	Message processing (ms)	Total (ms)
BAN Init.	14.58	11.81	3.68	30.08
BAN Resp.	14.03	2.86	1.62	18.52
ISO9798 Init.	13.07	35.784	23.30	72.16
ISO9798 Resp.	13.51	6.876	12.24	32.63
Kerb. Init. 1	22.63	0.83	0	23.47
Kerb. Init. 2	12.61	0.55	1.58	14.76
Kerb. Init. 3	2.23	3.34	0.94	6.52
Kerb. Resp. 1	19.28	0	0.41	19.69
Kerb. Resp. 2	10.81	3.379	1.67	15.87
Kerb. Resp. 3	5.25	11.41	3.59	20.26

# Conclusions and future work



- We have developed a framework for generating security protocol specifications
- We have validated the framework by constructing and executing over 38 specifications
- We intend to develop a tool for the automated construction of specifications



Thanks for your attention!