Automated Composition of Security Protocols

Genge Béla\textsuperscript{1}, Iosif Ignat\textsuperscript{2} and Haller Piroska\textsuperscript{1}

\textsuperscript{1}“Petru Maior” University of Târgu Mureș, Romania
\{bgenge, phaller\}@engineering.upm.ro

\textsuperscript{2}Technical University of Cluj Napoca, Romania
Iosif.Ignat@cs.utcluj.ro

August 28, 2009
Presentation overview

- Introduction & Motivation
- Proposed composition method
- Security protocol specification
- Experimental results
- Conclusions and future work
Basic concepts

- Security protocols are “communication protocols dedicated to achieving security goals” (Cremers and Mauw, 2005) such as confidentiality, integrity or availability.
- Over the last decade, researcher’s attention focused more on developing new security protocol design methods.
- One of the most popular methods is the *composition*: building new protocols from several existing smaller protocols.
Motivating scenario

- Service interconnection is a problem frequently encountered and addressed by many researchers today.
- There are many proposals dealing with the composition of service capabilities (Srivastava et al. 2003, Arpınar et al. 2004, Feenstra et al. 2007, ...)

![Diagram showing service interconnection](image-url)
Motivating scenario (cont’d)

- When using security protocols, the composition of services becomes a difficult task
- Existing solutions rely on using standard parameterized protocols implemented by every service
- ⇒ Services implementing new security protocols can not be composed with other services
One of the first proposals came from J.D. Guttman (Guttman, 2002), that used authentication tests as building blocks for multi-party authentication protocols.

Guttman’s authentication tests were later used by H.J. Choi (Choi, 2006) to develop a framework for constructing security protocols based on predefined protocols.

A. Datta et al. (Datta et al., 2007) propose a method where the composition process starts out from initial protocol equations and tries to reach the properties modeled by the final equations, corresponding to the composed protocol.

S. Andova et al. (Andova et al., 2008) propose a similar method to A. Datta, however, in this case the properties are verified automatically using an existing tool.
The solutions proposed by Guttman and Choi rely on predefined protocols, thus applying them in the composition of existing protocols is not possible.

The solutions proposed by Datta et al. and Andova et al. rely on the user to construct the security protocol equations.

The solution proposed by Andova et al. is semi-automatic because only the verification phase uses an automatic protocol verification tool.
Protocol model

- Basic sets:
  \( P, N, K, C, M \)

- Encryption functions:
  \( \text{FuncName ::= sk | pk | h | hmac} \)

- Terms:
  \( T ::= . | R | N | K | C | M | (T, T) | \{T\}_\text{FuncName(T)} \)

- Nodes and chains:
  \( \langle \sigma, t \rangle, \text{unde } \sigma \in \{+, -\}, t \in T \)
  \( \langle \pm t_1, \pm t_2, \ldots, \pm t_n \rangle \in (\pm T)^* \)

- Precondition-effect predicates:
  \( \text{CON\_CONF, CON\_KEYEXCHANGE} \ldots \in \text{PR\_CC} \)

- Term type predicates:
  \( \text{TYPE\_DN, TYPE\_KSYM} \ldots \in \text{PR\_TYPE} \)

- Participant and protocol models:
  \( \varsigma = \langle \text{prec, eff, type, gen, part, chain} \rangle \in \text{MPART} \)
  \( \bigcup \{\varsigma \mid \varsigma \in \text{MPART}\} \in \text{MPROT} \)
Composition of preconditions and effects

- Verifies that:
  - the knowledge required to run a given protocol, expressed through the form of precondition predicates, is available
  - the set of precondition and effect predicates is non-destructive

The first condition is verified by applying the \textit{PART\_PREC} predicate, defined for the $ctx \in T^*$ context as:

$$\textit{PART\_PREC}(ctx, eff_1, prec_2) = \begin{cases} 
\text{True,} & \text{if } eff_1 \subseteq prec_2 \cup \{ \cup \{ \text{CON\_TERM}(t) | t \in ctx \} \}, \\
\text{False,} & \text{otherwise.} 
\end{cases}$$

The second condition is verified by applying the \textit{PART\_NONDESTR} predicate, defined as:

$$\textit{PART\_NONDESTR}(eff_1, prec_2, eff_2) = \begin{cases} 
\text{True,} & \text{if } EF_1 \neq \text{CON\_CONF} \lor \text{if } EF_1 = \text{CON\_CONF} \land t_1 = t_2 \text{ then} \\
& \exists EF_2(t_2) : EF_2 = \text{CON\_CONF}, \\
& \forall EF_1(t_1) \in eff_1 \land \forall PR_2(t_2) \in prec_2, \\
\text{False,} & \text{otherwise.} 
\end{cases}$$
Composition of preconditions and effects (cont’d)

- In order to denote the PE-composition of two participant or two protocol models, we use the following operators:
  - For participant models:
    \[ \prec^{PE}_{\varsigma} : \text{MPART} \times \text{MPART} \rightarrow \text{MPART} \]
  - For protocol models:
    \[ \prec^{PE}_{\xi} : \text{MPROT} \times \text{MPROT} \rightarrow \text{MPROT} \]
  
- By applying the \[ \prec^{PE}_{\xi} \] operator on two protocol models \( \xi_1 \) and \( \xi_2 \), we have that:
  \[ \xi_1 \prec^{PE}_{\xi} \xi_2 \neq \xi_2 \prec^{PE}_{\xi} \xi_1 \]
Composition of protocol chains

- Verifies if attacks can be constructed on each protocol by using terms extracted from the other protocols
- Such a method was proposed in our previous work (Genge 2007, Genge 2008)
- The condition we proposed through the form of a proposition, would provide protocol independence, meaning that composed protocols for which this condition is satisfied would maintain their security properties
- In order to prove the correctness of the proposition, we constructed a canonical protocol model, based on the presented protocol model
In order to denote the PC-composition of participant and protocol models, we use the following operators:

- For participant models:
  \[ \prec_{\varsigma}^{\text{PC}} : \text{MPART} \times \text{MPART} \to \text{MPART} \]

- For protocol models:
  \[ \prec_{\xi}^{\text{PC}} : \text{M PROT} \times \text{MPROT} \to \text{MPROT} \]

By applying the \[ \prec_{\xi}^{\text{PC}} \] operator on two protocol models \( \xi_1 \) and \( \xi_2 \), we have that:

\[ \xi_1 \prec_{\xi}^{\text{PC}} \xi_2 \neq \xi_2 \prec_{\xi}^{\text{PC}} \xi_1 \]

If two protocol models can be composed PE and PC, then these can be composed using the following operator:

\[ \prec_{\xi}^{\text{C}} : \text{MPROT} \times \text{MPROT} \to \text{MPROT} \]
Security protocol specification

- In order to test the proposed composition method, we first constructed a specification.
- Each specification consists of several WSDL-S and OWL files: one WSDL-S and OWL file pair for each participant.
- Specifications were constructed according to the protocol model presented in this paper.

![Specification components diagram]
Part of Lowe’s BAN security protocol specification

- Key exchange protocol
- Requires previous knowledge on the shared key $K_{ab}$

\[
\begin{align*}
A, B : & \quad \text{principal} \\
K_{ab}, K'_{ab} : & \quad \text{symkey} \\
Na, Nb, N'b : & \quad \text{nonce} \\
\text{succ} : & \quad \text{nonce} \rightarrow \text{nonce} \\
1. & \quad A \rightarrow B : \quad A, Na \\
2. & \quad B \rightarrow A : \quad \{Na, K'ab, B\}K_{ab} \\
3. & \quad A \rightarrow B : \quad \{Na\}K'ab \\
4. & \quad B \rightarrow A : \quad Nb
\end{align*}
\]
Part of Lowe’s BAN specification (cont’d)

- Model protocol roles:

  ```xml
  <wssem:precondition name="Initiator"
  wssem:modelReference="http://...#Initiator"/>
  ```
Part of Lowe's BAN specification (cont’d)

- Model protocol roles:

  ```xml
  <wssem:precondition name="Initiator"
  wssem:modelReference="http://...#Initiator"/>
  ```

- Model preconditions:

  ```xml
  <wssem:precondition name="Part_A"
  wssem:modelReference="http://...#A"/>
  <wssem:precondition name="Part_B"
  wssem:modelReference="http://...#B"/>
  <wssem:precondition name="Key_AB"
  wssem:modelReference="http://...#Kab"/>
  ```
Part of Lowe’s BAN specification (cont’d)

- Model protocol roles:
  
  ```xml
  <wssem:precondition name="Initiator"
    wssem:modelReference="http://...#Initiator"/>
  ```

- Model preconditions:
  
  ```xml
  <wssem:precondition name="Part_A"
    wssem:modelReference="http://...#A"/>
  <wssem:precondition name="Part_B"
    wssem:modelReference="http://...#B"/>
  <wssem:precondition name="Key_AB"
    wssem:modelReference="http://...#Kab"/>
  ```

- Model initial terms and roles:

  ![Diagram of initial terms and roles]

  - InitialTerm
  - PartRole
  - A
  - B
  - Kab
  - Initiator
  - Respondent
  - ThirdParty
  - isOfType
  - UserName
  - isExtracted
  - Md_Name
  - isOfType
  - UserName
  - isExtracted
  - Md_Name
  - isOfType
  - SymmetricKey
  - isExtracted
  - Md_Keys
  - hasKeyParticipant
  - A
  - hasKeyParticipant
  - B
  - hasInitiator
  - A
  - hasRespondent
  - B
Model effects:

```xml
<wsem:effect name='SessionKeyExchange'
    wssem:modelReference='http://...#Session_key_exchange'/>
<wsem:effect name='SessionKey'
    wssem:modelReference='http://...#K'/>
```
Part of Lowe’s BAN specification (cont’d)

- **Model effects:**
  
  ```xml
  <wssem:effect name="SessionKeyExchange"
    wssem:modelReference="http://...#Session_key_exchange"/>
  <wssem:effect name="SessionKey"
    wssem:modelReference="http://...#K"/>
  ```

- **Model message 1:**
  
  ```xml
  <xsd:element name="Msg1">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="A" type="xsd:string"
          wssem:modelReference="http://...#Tx_m1"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  ```
Part of Lowe’s BAN specification (cont’d)

- **Model effects:**
  
  ```xml
  <wssem:effect name="SessionKeyExchange"
  wssem:modelReference="http://...#Session_key_exchange"/>
  <wssem:effect name="SessionKey"
  wssem:modelReference="http://...#K"/>
  ```

- **Model message 1:**
  
  ```xml
  <xsd:element name="Msg1">
  <xsd:complexType>
  <xsd:sequence>
  <xsd:element name="A" type="xsd:string"
  wssem:modelReference="http://...#Tx_m1"/>
  </xsd:sequence>
  </xsd:complexType>
  </xsd:element>
  ```

- **Model message 2:**
  
  ```xml
  <xsd:element name="Msg2">
  <xsd:complexType>
  <xsd:sequence>
  <xsd:element name="EncTerm1" type="xsd:base64Binary"
  wssem:modelReference="http://...#Rx_m1"/>
  </xsd:sequence>
  </xsd:complexType>
  </xsd:element>
  ```
Part of Lowe’s BAN specification (cont’d)

- Model message 3:

```xml
<xsd:element name="Msg3">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="EncTerm2" type="xsd:base64Binary">
        wssem:modelReference="http://..#Tx_m2"
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```
Part of Lowe’s BAN specification (cont’d)

- **Model message 3:**

  ```xml
  <xsd:element name="Msg3">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="EncTerm2" type="xsd:base64Binary">
          <wssem:modellReference="http://...#Tx_m2"/>
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  
  - **Model message 4:**

  ```xml
  <xsd:element name="Msg4">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="Nb" type="xsd:base64Binary">
          <wssem:modellReference="http://...#Rx_m2"/>
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  ```
Part of Lowe’s BAN specification (cont’d)

- Model comm terms:

```
CommunicationToken

  SentTerm
    Tx_m1
      Tx1_A
        isExtracted
        isBeforeOf
        Tx1_Na
    Tx_m2
      Tx1_Na
        isExtracted
        Na
    Tx2_Enc1
      isExtracted
      Enc1

  RecvdTerm
    Rx_m1
      Rx1_Enc1
        isDecrypted
        Enc2
        Nb
    Rx_m2
      Rx2_Nb
        isStored
```
Part of Lowe’s BAN specification (cont’d)

- Model generated terms:
Part of Lowe’s BAN specification (cont’d)

- **Model generated terms:**
  
  - `Na`
  - `Enc1` generatedTerm
  - `Enc1_Na`
  - `isOfTyp` Random
  - `hasLength` Len_128
  - `hasExtracted` `isExtracted` `Na`

- **Model discovered terms:**
  
  - `Enc2` discoveredTerm
  - `Enc2_Na`
  - `Enc2_K`
  - `Enc2_B`
  - `isOfTyp` SymmEncrypted
  - `hasSymmAlgorithm` AES
  - `hasKey` `K`
  - `hasAlgorithmMode` CBC

- **Additional terms:**
  
  - `Nb`
  - `K`
  - `isOfTyp` SymmKey
  - `hasKeyParticipant` A
  - `hasKeyParticipant` B
  - `isBeforeOf` `Enc2_K`
  - `isBeforeOf` `Enc2_B`
Part of Lowe’s BAN specification (cont’d)

- **Model generated terms:**

  - Na
  - Enc1
  - Enc1_Na

- **Model discovered terms:**

  - Enc2
  - Enc2_Na
  - Enc2_K
  - Enc2_B

- **Model previous terms:**

  - PreviousTerm
  - Prev_Nb
  - isOf Type
  - Random
  - isExtracted
  - Md_PrevValues

- **Model discovered terms:**

  - Nb
  - K
  - isOf Type
  - Symm Key
  - hasKey Participant A
  - hasKey Participant B
  - hasFreshness
  - Prev_Nb
## Composition results

**Table:** Protocol composition results

<table>
<thead>
<tr>
<th>Protocol 1</th>
<th>Protocol 2</th>
<th>Precondition-Effect (S1/S2)</th>
<th>Protocol-Chain (S1/S2)</th>
<th>Validation: Scyther</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lowe-B</td>
<td>ISO9798</td>
<td>N/Y</td>
<td>Y/Y</td>
<td>Y/Y</td>
</tr>
<tr>
<td>Lowe-B</td>
<td>X509v1</td>
<td>N/N</td>
<td>Y/Y</td>
<td>Y/Y</td>
</tr>
<tr>
<td>ISO9798</td>
<td>X509v1</td>
<td>Y/Y</td>
<td>Y/Y</td>
<td>Y/Y</td>
</tr>
<tr>
<td>ISO9798</td>
<td>X509v1c</td>
<td>Y/Y</td>
<td>Y/Y</td>
<td>Y/Y</td>
</tr>
<tr>
<td>X509v1</td>
<td>X509v1c</td>
<td>Y/Y</td>
<td>Y/Y</td>
<td>Y/Y</td>
</tr>
<tr>
<td>X509v1</td>
<td>X509v1c</td>
<td>Y/Y</td>
<td>Y/Y</td>
<td>Y/Y</td>
</tr>
<tr>
<td>BAN-RPC</td>
<td>Lowe-B</td>
<td>Y/Y</td>
<td>N/N</td>
<td>N/N</td>
</tr>
<tr>
<td>L-D-S</td>
<td>K-Cv1</td>
<td>Y/N</td>
<td>N/N</td>
<td>N/N</td>
</tr>
<tr>
<td>K-Cv1</td>
<td>K-Cv2</td>
<td>Y/Y</td>
<td>Y/Y</td>
<td>Y/Y</td>
</tr>
<tr>
<td>L-D-S</td>
<td>Kerbv5</td>
<td>Y/Y</td>
<td>N/N</td>
<td>N/N</td>
</tr>
<tr>
<td>Lowe-Kerb</td>
<td>Neuman-S</td>
<td>Y/N</td>
<td>N/N</td>
<td>N/N</td>
</tr>
<tr>
<td>H-N-S</td>
<td>Neuman-S</td>
<td>Y/Y</td>
<td>Y/Y</td>
<td>Y/Y</td>
</tr>
<tr>
<td>Needh-S</td>
<td>X509v1</td>
<td>Y/N</td>
<td>Y/N</td>
<td>Y/Y</td>
</tr>
<tr>
<td>L-N-S</td>
<td>ISO9798</td>
<td>Y/N</td>
<td>Y/N</td>
<td>Y/Y</td>
</tr>
<tr>
<td>Otway-R</td>
<td>Lowe-B</td>
<td>Y/N</td>
<td>Y/N</td>
<td>Y/Y</td>
</tr>
<tr>
<td>SPLICE</td>
<td>Needh-S</td>
<td>Y/Y</td>
<td>Y/Y</td>
<td>Y/Y</td>
</tr>
<tr>
<td>TMN</td>
<td>Andr-RPC</td>
<td>Y/N</td>
<td>Y/N</td>
<td>Y/Y</td>
</tr>
<tr>
<td>Y-L</td>
<td>K-Cv1</td>
<td>Y/Y</td>
<td>N/N</td>
<td>N/N</td>
</tr>
</tbody>
</table>
Composition of video services

Service Capabilities and Security Protocol Composition

- Name Service
- Client Terminal
- Authorization and Composition Service
- Specification Service
- Video Service
- Save Service

SAVED DATA
Performance of composition modules

- Composition time of 4 resources
- Composition time of 50 resources
Composition of video services (cont’d)

Total accessing time of composed resources

- Accessing time for up to 50 resources
We developed a method for the composition of security protocols.

The novelty of our approach is the fact that it provides a syntactical verification of the involved protocols, that makes it appropriate for on-line automated composition applications.

The proposed method was used in the process of automated composition of security protocols for Web services.

As future work, we intend to use the proposed composition method in the design process of new protocols for Web services.

This would allow us to implement more complex protocols, such as TLS, currently used as a binary security protocol, in XML message format.
Thanks for your attention!

Questions?