Towards a Semantic of XML Signature
-How to Protect Against XML Wrapping Attacks

Sebastian Gajek, Lijun Liao, Jörg Schwenk
Horst-Görtz-Institut
Ruhr-University Bochum, Germany

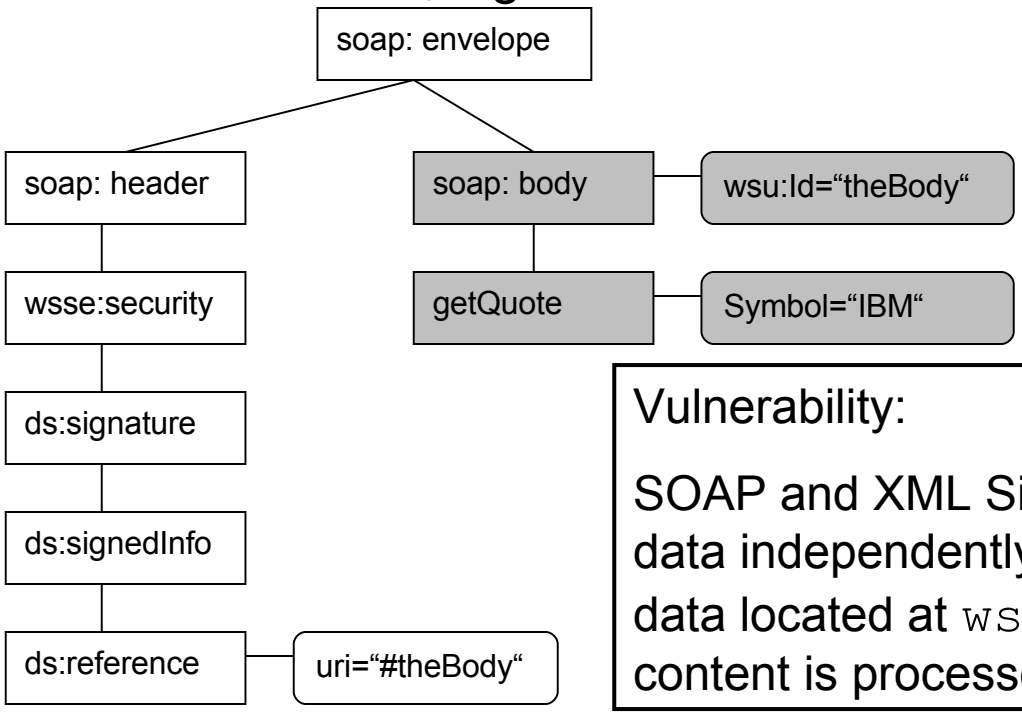
Presented by
Michael McIntosh
Java and Web Services Security Group
Security, Privacy, and Extensible Technologies Department IBM Research

Overview

- XML Wrapping Attacks (McIntosh and Austel 2005)
- Receiver-side Protection:
  - Strict Filtering
  - Returning a Spanning Tree
  - Returning Location Hints
- Sender-side Protection:
  - XPath
- Towards a formal Semantic for XML Signature

XML Wrapping Attacks (McIntosh and Austel 2005)

- The original document: The SOAP Body is signed and referenced by a `wsu:id` attribute; signature verification returns Boolean value.

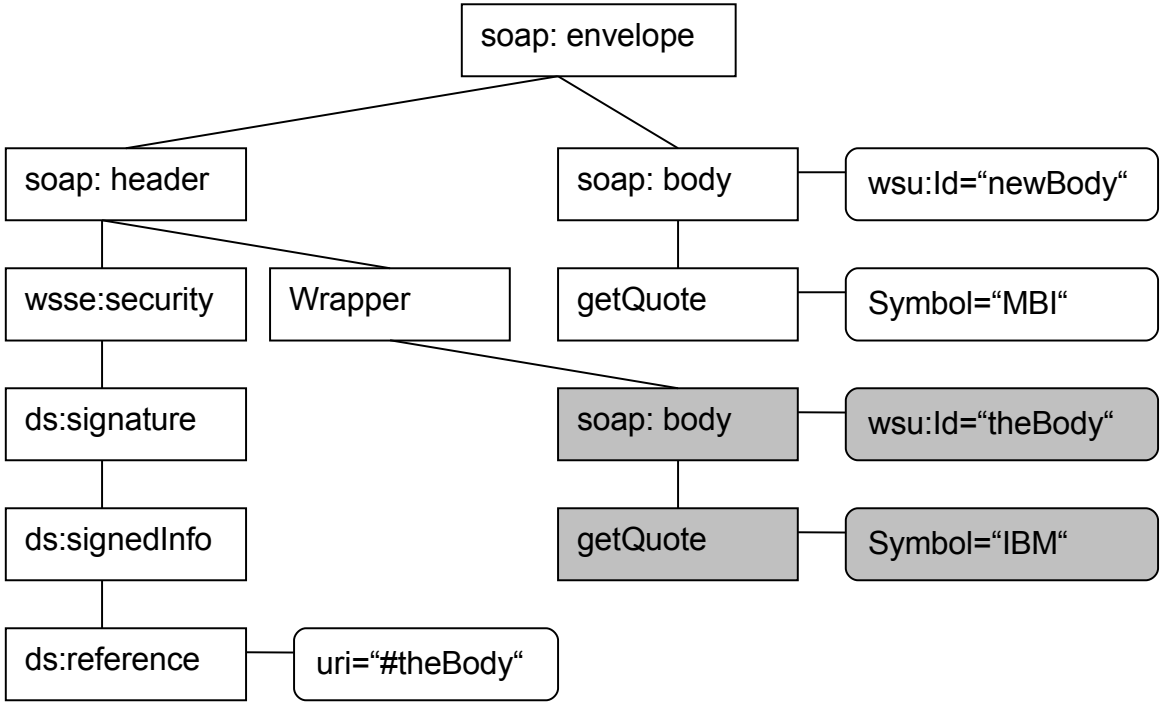


Vulnerability:

SOAP and XML Signature logics process data independently. That is, when signed data located at `wsu:id` is valid then the content is processed by SOAP engine.

XML Wrapping Attacks (McIntosh and Austel 2005)

- The modified document: The same data is signed and referenced by a `wsu:id` attribute, but the SOAP Body has changed.



XML Wrapping Attacks (McIntosh and Austel 2005)

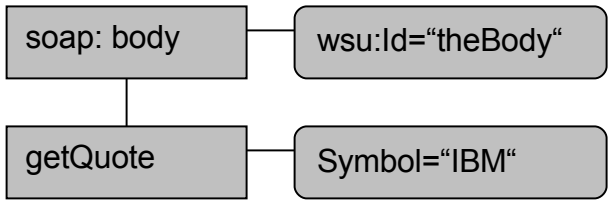
Summary

- Wrapping Attacks do not change the semantics of XML signatures using `wsu:id`
- However it would be useful to be able to detect such modification
- For other signature formats (OpenPGP, PKCS#7) it is sufficient to return a Boolean value after verification; for XML Signature, this is no longer the case

Receiver-side Protection: Strict Filtering

Solution 1: Business Logic only gets the signed data

- Signature verification is located as a filter between the network and the Business Logic
- Effect: Only the following XML fragment is forwarded to the Business Logic

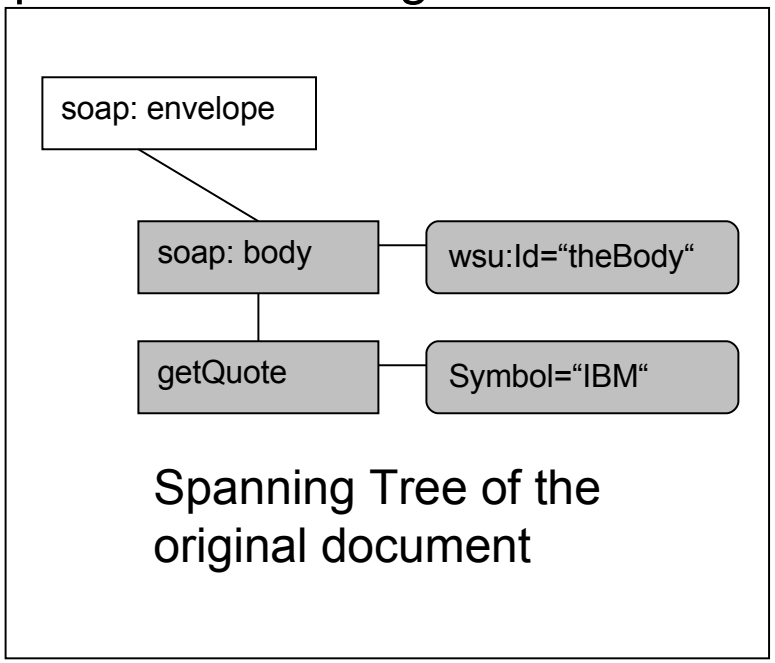


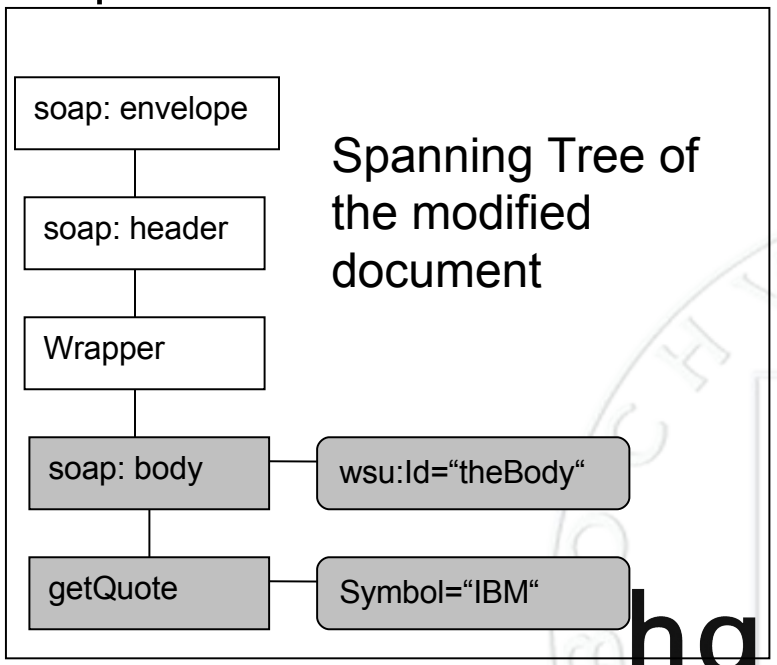
- Problems: Transform within each <Reference>-Element may result in non-XML data.

Receiver-side Protection: Returning a Spanning Tree

Solution 2: The signature verification function returns the signed data plus all elements up to the root of the document

- Effect: The Business Logic can compare the actual (vertical) position of the signed data with its expectations

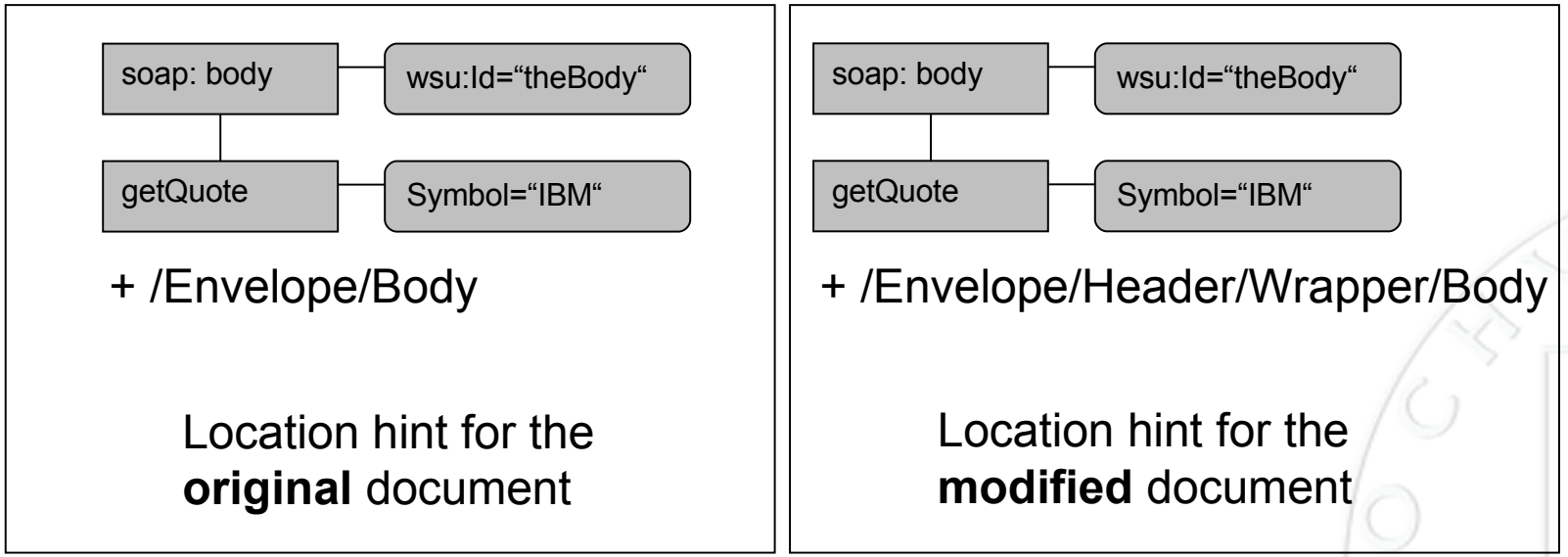




Receiver-side Protection: Returning Location Hints

Solution 3: The signature verification function returns the signed data plus an absolute XPath describing its (vertical) position

- Effect: The Business Logic can compare the actual (vertical) position of the signed data with its expectations



Sender-side Protection: XPath

Solution 4: The sender fixes the position of the signed data via XPath

- Either XPath transform (Version 1 or 2)
- or XPointer argument in URI (not yet tested)

Effect: Any modification to the document changing the position of the signed data will be detected

Future Work: Towards a formal Semantic for XML Signature

The XML Signature standard contains useful (e.g., XPath) and dangerous (e.g., XSLT) transforms.

A formal semantics should help understand which parts of an XML document are protected by the signature. The most important part is to understand how the different types of URIs and XPath transform influence the protected parts.

Known formal semantics for XPath in XSLT are not clear enough because they only map to (unordered) XML nodesets.

A future semantic for XPath/URIs in XML Signature should map into (mathematical) trees/forests, along the lines of solution 2.

A Hybrid Solution

- Use an IDREF in the Signature Reference
- Use a Transform
  - With Path (XPath syntax) from Root to Referenced Element
  - Processing verifies Path
    - Output = Input if Path matches
    - Output != Input if Path does not match
- Assumes implementation has access to equivalent of Node::getParent