How to Break XML Encryption – Automatically

Dennis Kupser, Christian Mainka, Jörg Schwenk, Juraj Somorovsky

Horst-Görtz Institute for IT-Security
Ruhr-University Bochum

Christian.Mainka@rub.de
Twitter: @CheariX
Quick Introduction
XML-based Web Service

Client

Envelope

Body

getPrime

Server

Envelope

Body

thePrime

11
Motivation – XML Security

• Web Services: Method for machine-to-machine communication over networks

• Used in commerce, finance, government, military, ...

• XML-based message format
Motivation – XML Security

• SSL / TLS: transport-level security

• Messages secured only during transport!
Motivation – XML Security

• Message level security

• Security applied directly on the messages
• No need for SSL / TLS
• Realized using XML Signature, XML Encryption
XML Security

- W3C Standard for securing XML messages
- XML Encryption: protects confidentiality
- XML Signature: protects authenticity and integrity
- Very flexible

```xml
<PaymentInfo>
  <Name>John Smith</Name>
  <CreditCard Limit='5,000'>
    <Number>4019 ...5567</Number>
    <Issuer>Example Bank</Issuer>
    <Expiration>04/02</Expiration>
  </CreditCard>
</PaymentInfo>
```
XML Encryption

Hybrid encryption scheme

1. Asymmetric encryption / decryption
   - Algorithm="...#rsa-1_5"
   - URI="#enc"

2. Symmetric encryption / decryption
   - Algorithm="...#aes128-cbc"
Attacks on XML Encryption

• Attacks on EncryptedData
  – How to Break XML Encryption.
    Tibor Jager, Juraj Somorovsky. CCS 2011

• Attacks on EncryptedKey
  – Bleichenbacher’s Attack Strikes Again:
    Breaking PKCS#1 v1.5 in XML Encryption.
    Tibor Jager, Sebastian Schinzel, Juraj Somorovsky. ESORICS 2012
Attack Concept: *Validity Oracle*

- Attacker needs to send arbitrary ciphertexts!
Countermeasures?
Countermeasures – XML Signature?

Attacker able to create valid signatures?
Countermeasures – XML Signature?

XML Signature Wrapping

Signature
  - Reference
    - URI="#body"
  - Body
    - Id="body"
  - EncryptedData
    - Id="enc"
  - CipherData

EncryptedKey
  - DataReference
    - URI="#oracle"
  - Body
    - Id="body"
  - EncryptedData
    - Id="enc"
  - CipherData

Signature validation ✅

Decryption ✅

McIntosh and Austel. XML Signature Element Wrapping attacks, 2005
Encryption Wrapping attack: WS-Security Policy says, what must be encrypted... but it says not, what must not be encrypted.
WS-Attacker Implementation
Automated Attack Workflow

Encrypted XML

Detection Phase

Avoid Phase

Attack Phase

Identify Security Elements
Detection Phase (Offline)

Diagram:
- Envelope
  - Header
  - Security
    - EncryptedKey: URI="#a"
    - Signature
      - Signature: URI="#b"
      - Signature
      - Timestamp: Id="c"
  - Body: Id="b"
  - EncryptedData: Id="a"
Automated Attack Workflow

- Detection Phase
  - Identify Security Elements
  - Signed Timestamp?

- Avoid Phase

- Attack Phase

- Knowledge Pool

Encrypted XML
Detection Phase (Offline)
Automated Attack Workflow

- Detection Phase
- Avoid Phase
- Attack Phase

Identification of Security Elements

- Encrypted XML

Knowledge Pool

Signed Timestamp?

- yes
  - XSW

- no
Applying XSW - Complexity

No Attribute

Same Attribute

Position 1

Position 2

Position n
Automated Attack Workflow

Identify Security Elements

Signed Timestamp?

Signed Encrypted Element

Identify Oracle

XSW

no

yes

no

yes

fail

Detect Phase

Avoid Phase

Detection Phase

XEW

yes

no

fail

Encrypted XML
Identify Oracle

- Map Server Responses to „valid“ or „invalid“

Chosen ciphertext (IV’, C₁)
<ok/>
Chosen ciphertext (IV”, C₁)
<failure/>

- Implementation dependent!
Automated Attack Workflow

Encrypted XML → Identify Security Elements → Signed Timestamp?
- no → Signed Encrypted Element
- yes → xSW

Signed Encrypted Element
- yes → xSW
- no → Identify Oracle

Identify Oracle
- yes → Apply Attack
- no → XEW

XEW
- no → Avoid Phase
- yes → fail

Avoid Phase
- no → fail
- yes → Detect Phase

Detect Phase
- no → Decrypted XML
- yes →

Attack Phase

Knowledge Pool
## Results

<table>
<thead>
<tr>
<th>System</th>
<th>Asymmetric Attack</th>
<th>Symmetric Attack</th>
<th>Countermeasures applicable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apache Axis2 1.6.2</td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Apache CXF 2.7.10</td>
<td>✓</td>
<td>✓</td>
<td>yes</td>
</tr>
<tr>
<td>Axway Gateway 7.3.1</td>
<td>✓</td>
<td>✓</td>
<td>yes</td>
</tr>
<tr>
<td>IBM Datapower XI50</td>
<td></td>
<td>✓</td>
<td>yes</td>
</tr>
<tr>
<td>Microsoft WCF</td>
<td></td>
<td></td>
<td>yes</td>
</tr>
</tbody>
</table>
Conclusion

• XML – especially XML Security – is complex
• WS-Attacker can be used to test Web Service implementations automatically
• Our approach is applicable to other scenarios
  – SAML, JSON, ...
• Preffer authenticated encryption (AES-GCM instead of AES-CBC)

Q&A?